Tuntisuunnitelmapankki

Osallistujien tekemiä tuntisuunnitelmia kurssilta
Algoritmisen ajattelun kehittäminen (LUMATIKKA-ohjelma)

27. joulukuuta 2022

Sisällys

Otsikot ovat klikattavia.

1 Alkusanat

2 Musiikkia 4-vuotiaiden kanssa, varhaiskasvatus (Mirja Leppiviita)

3 Kuvanetsintää 4–5-vuotiaden kanssa, varhaiskasvatus (Nimetön)

4 Ohjelmointia kehollisesti 4–6-vuotiaden integroidulle erityisryhmälle, varhaiskasvatus (Nimetön)

5 Robottileikkejä 4–6-vuotiaille, varhaiskasvatus (Raisa Miettinen)

6 Ohjelmointia liikunnallisin ja leikillisin keinoin 5-6-vuotiaille, varhaiskasvatus (Nina Parviainen)

7 Ohjelmoinnillisen ajattelun harjoittelua musiikin ja liikunnan avulla, varhaiskasvatus (Johanna Tuupainen)

8 Mikä tekee koirasta koiran? - harjoituksen sovellus, varhaiskasvatus (Nimetön)

9 Kolmiosainen ohjelmointituokioiden kokonaisuus, varhaiskasvatus (Nimetön)

10 Geometriaa ja robottijumppaa, varhaiskasvatus (Nimetön)

11 Beebot ohjelmointi ja matematiikan toiminnallisia tehtäviä, esiopetus (Satu Porkka)

12 Ohjelmointia esikoululaisille, esiopetus (Heli-Similä Saukkonen)

13 Ohjelmointia esioppilaiden kanssa, esiopetus (Nimetön)

14 Koodauksen alkeet, esiopetus (Tiina Lähteenaro)

15 Paikkakäsitteiden ja ongelmanratkaisutaitojen harjoittelua BeeBot-roboteilla, esiopetus (Nimetön)

16 Ohjelmointia toiminnallisesti robottileikkien avulla, esiopetus (Anna Långström)

17 BeeBot-robotteihin tutustumista, esiopetus (Nimetön)

18 Ohjelmointiin tutustumista toiminnallisesti , esiopetus (Nimetön)

19 Algoritmileikkejä, esiopetus (Nimetön)

20 Ohjeiden ja käskyjen harjoittelua, esiopetus (Nimetön)

21 Ongelmia ja ohjelmointia, esiopetus (Tiia Leppänen)

22 Kirjainten kertaaminen BeeBot-robottien avulla, esiopetus (Nimetön)

23 Koodausta ekalle luokalle ilman koneita ja esimerkkinä ohjeiden anto, 1.lk (Salla Liusjärvi)

24 Lukemisen ja äänteiden harjoittelua ohjelmoinnin avulla, 1.lk (Hanna Hämäläinen)

25 Kymmenylityksen harjoittelua ohjelmoinnin avulla, 1.lk (Kaija Karlsson)

26 Ohjelmointia LEGO-roboteilla, 1.lk (Daria Korkka)

27 Äidinkieltä ja ohjelmointia, 1.lk (Nimetön)

28 Kymppiparien harjoittelu ohjelmoinnin avulla, 1.lk (Taija Caselius)

29 Tupiin jako ja Kalkaroksen liemitunti -pakohuonepeli, 1.-2. lk (Mirva Slunga)

30 BeeBot ja suuntien harjoittelu, 2.lk (Nimetön)

31 Unplugged ja Lightbot, 2.lk (Sini Hostikka)

32 Aarteenmetsästystä, 2.lk (Nimetön)

33 Kahden kertotaulua ja ohjelmointia, 2. lk (Hanne Kajasviita)

34 Kertotaulun harjoittelua BeeBot-roboteilla, 2. lk (Nimetön)

35 Ohjelmointi ja käsityöt: sinivuokkoliinan ompeleminen, alkuopetus (Nimetön)

36 Ohjelmointi ja käsityöt: paperilennokki, helminauha ja kirjonta, alkuopetus (Nimetön)

37 Ohjelmoinnin alkeet pistetyöskentelynä, alkuopetus (Nimetön)

38 Salainen polku: Ohjelmoinnin alkeita toiminnallisesti ilman tietokoneita, alkuopetus (Nimetön)

39 Ohjelmointia nuolikorteilla, alkuopetus (Nimetön)

40 Ohjelmoinnin alkeet ScratchJr:lla, alkuopetus (Nimetön)

41 Ympäristötieto, ihminen ja ohjelmointi, 2.–3. lk (Nimetön)

42 Yhteistoiminnallista tutustumista algoritmikäsitteeseen, 3.lk (Nimetön)

43 Ohjelmoinnin harjoituksia ilman tietokonetta, 3.lk (Kristiina Länsiö)

44 Liikuntaa ja ohjelmointia, 3.lk (Nimetön)

45 Ohjelmointi-suunnistus Seppo-pelialustalla, 3. lk (Mira Christiansen)

46 Ohjelmointia ruudukossa ja kertolaskujen kertaus, 3. lk (Nimetön)

47 Oma tarina elämään Scratchin avulla, 3. lk (Nimetön)

48 ScratchJr ja liikennemerkit, 3. lk (Nimetön)

49 Karttamerkkien harjoittelua ohjelmoinnin avulla, 3. lk (Nimetön)

50 Toiminnallinen tunti ohjelmoinnillisen ajattelun harjoitteluun, 3. lk (Nimetön)

51 Tapahtumasarjan ohjelmointi Scratchilla, 3.–4. lk (Anna-Maija Karttunen)

52 Koordinaatiston harjoittelu laivanupotus-pelin avulla sekä Scratch-projekti, 3.-4. lk (Maija Takalo)

53 Ohjelmoinnin alkeita: BeeBot ja Code.org, 3.–6.lk (Nimetön)

54 Visuaalisen ohjelmoinnin harjoittelu, 3.–6. lk (Nimetön)

55 Koodilukko: Ongelmanratkaisua silmukalla ja listalla, 3.–6. lk (Teemu Korhonen)

56 Algoritmista ajattelua liikkuen ja ilman tietokoneita, 4.lk (Nimetön)

57 BeeBot-ohjelmointitehtävän suunnittelu, 4.lk (Nimetön)

58 Koordinaatisto ja ohjelmointi, 4.lk (Jonna Hakala)

59 Algoritmeja toiminnallisesti, 4.lk (Anitta Leipälä)

60 Pistetyöskentelyä Scratchin ja BeeBottien parissa, 4.lk (Emmi Mykkänen)

61 Ohjelmoinnin suunnitelma, 4.lk (Suvi Kaukovalta)

62 Ohjelmointikokeiluja Scratchilla, 4.lk (Nimetön)

63 Scratch-ohjelmointiin tutustumista, 4.lk (Nimetön)

64 Algoritminen ajattelu kasvin tunnistuksessa, 5.lk (Timo Hartus)

65 BeeBot-kierrätyspeli, 5.lk (Nimetön)

66 Scratch-projekteja pienryhmissä, 5. lk (Nimetön)

67 Ohjelmointijakson aloitustunnit, 5. lk (Mari Lehikoinen)

68 Tarinan tekeminen Scratchilla, 5. lk (Tiina Niiranen)

69 Kuplalajittelu, 5. lk (Nimetön)

70 Animaation tekeminen Scratchilla, 5. lk (Nimetön)

71 Pelin ohjelmointi Scratchilla, 5.–6. lk (Veera Salminen)

72 Ohjelmoinnin kokonaisuus ScratcJr:lla ja Scratchilla, 5.-6. lk (Nimetön)

73 Ohjelmointia kotitalouden opetukseen , 6.lk (Mari Muhonen)

74 Historian opetus ja algoritmisen ajattelun kehittäminen, 6. lk (Nimetön)

75 Binääriluvut ja taikatemppuja, 6. lk (Krista Taipalvesi)

76 Videoita, pelejä ja Scratch, 6.-7.lk (Päivi Viinikka)

77 Scratch-projekti tutun tarinan pohjalta, alakoulu (Maria Jussila)

78 Tuntisuunnitelmat ohjelmointitunneille Scratchillä, alakoulu (Outi Liukkonen)

79 Ohjelmoinnin alkeet -kokonaisuus, alakoulu (Nimetön)

80 Ruotsin alkeita Scratchissa, alakoulu (Nimetön)

81 Algoritmisen ajattelun harjoittelua eri oppiaineissa, alakoulu (Nimetön)

82 Halloween-teemaista ohjelmointia BeeBot-roboteilla, alakoulu (Nimetön)

83 Tarinan kuvittaminen Scratchilla, alakoulu (Nimetön)

84 Micro:bit alkeet käsitöissä, 6.-8. lk (Tiina Torvinen)

85 Kertolaskupeli Microbitilla, 7.lk (Jani Heinonen)

86 Laskutoimituksia luvuilla, 7.lk (Aija Elo)

87 Ohjelmoinnin oppikokonaisuus, 7.lk (Hanna-Leena Hämäläinen)

88 Python Turtle ja monikulmiot, 7.lk (Tomi Hautakangas)

89 Geometrian kuvioita ohjelmoimalla, 7.lk (Riitta Kotilainen)

90 Koordinaatiston kertaamista ohjelmoinnin avulla, 7. lk (Nimetön)

91 Ohjelmointia Pythonilla, 7. lk (Svetlana Kallonen)

92 Kilpikonnagrafiikkaa Pythonilla, 7. lk (Tiina Kauvosaari)

93 Ehtorakenteiden harjoittelua Pythonilla, 7.-8. lk (Nimetön)

94 Geometrisia kuvioita Turtlen avulla, 7.-9. lk (Anna Norrkniivilä)

95 Python, 8.lk (Liliana Peuhkuri)

96 Luvun muodostaminen (Scratch-ohjelmontia), 8. lk (Nimetön)

97 Python-ohjelmoinnin alkeet, 8. lk (Nimetön)

98 Kappaleesen vaikuttavia voimia fysiikan simulaationa, 8.-9.lk (Reima Halmetoja)

99 Pythagoraan lause, 8.-9. lk (Nimetön)

100 Python-ohjelmointia 9.lk (Nimetön)

101 Klassinen ja tilastollinen todennäköisyys Pythonilla, 9. lk (Saana Saajoranta)

102 Peliohjelmointia Pythonilla, 9. lk (Ari Virtanen)

103 Karkausvuosi-ohjelmointia, 9. lk (Nimetön)

104 Python-ohjelmointia Tie koodariksi -sivuston avulla, 9. lk (Johanna Vaurasalo)

105 Micro:bit, yläkoulu (Sanna Toivanen)

106 Ehtolauseiden harjoittelua Pythonilla, yläkoulu (Heini Soppi)

107 Ohjelmoinnin opiskelu aoe.fi-sivuston materiaaleja hyödyntäen, yläkoulu (Tero Toivanen)

108 Alkuluvut ja Eratostheneen seula, yläkoulu (Nimetön)

109 Kilpikonnagrafiikka (Python Turtle), yläkoulu (Anna Leinonen)

110 Alkulukujen etsintää Python-ohjelmoinnin avulla, yläkoulu (Nimetön)

111 Liikunnallista ohjelmointia, yläkoulu (Nimetön)

112 Sovelluksen ohjelmointi code.org:n App Labissa, yläkoulu/lukio (Suula Arppe)

113 Todennäköisyyksien laskentaa Pythonilla, yläkoulu/lukio (Annukka Rautiola)

114 **Matka-aikojen laskemista Pythonilla, yläkoulu/lukio** (Katariina Piri)

115 **Pitkän matematiikan todennäköisyyslaskentaa ohjelmoimalla, lukio** (Nimetön)

116 **Eukleideen algoritmi, lukio** (Milla Hirvonen)

117 **Geometrinen todennäköisyys, lukio** (Miika Karpin)

118 **Pythonin alkeiden harjoittelua, lukio** (Pirjo-Riitta Elo)

119 **Python-ohjelmointia, lukio** (Maiju Mäenpää)

120 **Let's play Countdown!, toisen asteen ammatilliset opinnot** (Erik Eriksson ja Suvi Kääriälä)

1 Alkusanat

Tämän materiaalikokoelman tuntisuunnitelmat on laadittu kurssitöinä täydennyskoulutuskurssilla *Algoritmisen ajattelun kehittäminen*, joka on osa LUMATIKKA-ohjelmaa. Kokoelmassa ovat mukana vain ne tuntisuunnitelmat, joiden laatijat ovat antaneet luvan laatimansa materiaalin julkaisemiseen. Kokoelmaa laajennetaan, kun uusilla kurssi-iteraatioilla syntyy lisää materiaalia. PDF-muotoiset tuntisuunnitelmat on sisällytetty sellaisinaan, mahdolliset suunnitelman omat kansilehdet poistaen, ja muissa muodoissa olleet suunnitelmat on kokoelmaan sisällyttämistä varten muunnettu PDF-muotoon niiden sisältöä kuitenkaan muokkaamatta.

Tehtävänantona oli suunnitella noin kahden oppitunnin mittainen kokonaisuus ja dokumentoida se siten, että sen perusteella toinen opettaja tai esimerkiksi sijainen pystyy toteuttamaan tunnin luokassaan. Kohderyhmän, käytetyt välineet ja teknologiat sekä kokonaisuuteen mahdollisesti nivoutuvat muut teemat sai kukin valita vapaasti. Kunkin suunnitelman kirjoittaja, kohderyhmä ja käytetty teknologia on merkitty suunnitelman kansilehdelle.

Musiikkia 4-vuotiaiden kanssa, varhaiskasvatus

Mirja Leppiviita, CC BY-SA 4.0

2.4.2021

Valitsin koodikieleksi jo olemassa oleven kielen, nuotit. Ajatuksen sain Läps taps-kielestä (Ohjelmoinnin ABC)

Olemme laulaneet lasten kanssa talven ajan laulua talitintistä:

Talitintti hakkaa I I I Z (3 taputusta ja tauko)

jäistä talipakkaa I I I Z

Kylmä on ei takkaa I I I Z

pakkanen ei lakkaa I I I Z

...

Osa lapsista suoriutui hienosti, osalle oli vaikeaa. Ajattelin konkretisoida asian vielä visuaalisesti nuotti- ja tauko-korteilla.

Ensimmäisen tunnin tavoitteet:

- Yhteinen rytmi, kuin sydämen syke.
- Ääni ja tauko -vaihtelut.
- Myöhemmin myös nopea-hidas -vaihtelut.
- Lapsi osallistuu ja yrittää tehdä ohjeen mukaan.

Välineet ja menetelmät:

- Tuttu talitintti laulu.
- Nuotti ja taukokortit, vihreitä kortteja (väh 12kpl) joissa ¼ nuotin kuva ja sinisiä (väh 4) joissa ¼ tauon kuva. Kortteja tarpeeksi
- Kysyn, onko joku nähnyt korttien kuvia ja mitä ne mahtaa tarkoittaa?
- Käydään yhdessä kortit läpi. (TAA ja TAUKO)
- Laitan talitintti rytmin valmiiksi: TAA TAA TAA TAUKO Ja taputamme sen. Onnistuttuamme laulamme talitintti laulun, jonka jälkeen laitan peräkkäin yhteensä 4 samanlaista tahtia.
- Pyydän jokaista lasta laittamaan vuorollaan 8 korttia haluamaansa järjestykseen ja taputamme rytmin. Tallennamme rytmit ja kuvat iPadille.
- Kerron lapsille meidän yhdessä olevan rytmikone, jokaa lukee annetun rytmin ja toimii sen mukaisesti. Siksi yhteinen pulssi on tärkeä.
- Mietimme mikä rytmi (ostinato) kuulostaa kivalle ja minkä päälle olisi helppo laulaa
- Lopuksi yritämme tallentaa yhden rytmin Garage Band -sovellukseen ja laulaa lauluraidan siihen päälle

Toinen tunti:

Tavoitteet samat, lisäksi lapsi ymmärtää että rytmejä voi tallentaa symboleilla ja että rytmit toistuvat samanlaisina uudelleen soitettuna. Tutustutaan TI-TI nuottiin.

- Muistellaan edellisen kerran tekemisiä korttien avulla ja kuunnellaan iPadille tallennettuja rytmejä.
- Esittelen TI-TI -kortin. Miten se eroaa TAA -kortista?
- Teen korteista rytmin TAA TI-TI TAA TAA ja toistamme sitä useaan kertaan.
- Kuuntelemme pianoversiona teeman Beethovenin seitsemännestä sinfoniasta 11 allegretto, jonka läpi TAA TI-TI TAA TAA kulkee ja kyselen huomasivatko lapset sen?
- Lapset saavat taas tehdä omia rytmejä ja taputetaan ne kaikki läpi. Todetaan tehneemme rytmejä, joita kuka tahansa voi soittaa annetun koodin perusteella.

Tässä suunnitelmassa voi olla, lapsiryhmästä riippuen, aineksia useampaankin tuntiin.

Tuttuja rytmejä on hyvä käyttää myös jumppahetkellä ja kuvallisessa ilmaisussa/askartelussa voi tehdä vaikka rytmikoneen.

Saa nähdä mitä huomioita lapset tekevät, kun näiden hetkien jälkeen taputetaan lasten nimiä ja muita sanarytmejä.

Kuvanetsintää 4–5-vuotiaden kanssa, varhaiskasvatus

30.4.2022

Algoritmisen ajattelun harjoittaminen toteutetaan metsässä kuvanetsintänä sekä kuvallisen ohjeen rakentamisena pienryhmissä 4–5-vuotiaiden lasten kanssa. Ajatuksena on tuoda lasten omaan arkeen ja pukeutumistilanteisiin hieman uutta näkökulmaa ja kuvia. Viime aikoina juuri nuo tilanteet ovat olleet osalle lapsista haastavia ja turhauttavia. Ajatuksena on saada tukea ja hyötyä koko ryhmän siirtymätilanteisiin ilman, että joku yksittäinen lapsi aiheesta leimaantuisi tai enempää turhautuisi. Toiminta toteutetaan kahtena päivänä. Ensimmäisellä kerralla olemme mestässä n. 45 min ja toinen kerta on kestoltaan n. 30 min.

Toiminta alkaa pienellä tarinalla ja seuraavaksi pienryhmä löytää symbolien (neliö, kolmio, ympyrä, suorakulmio) avulla lähtöpiste kuvien etsinnälle ja sen jälkeen etsitään kuvan kautta kadonneita kuvia metsästä. Seuraavana päivänä kootaan kuvat oikeaan järjestykseen sarjaksi. Yhdessä rakennettu ohjeistus pukeutumistilanteita varten otetaan käyttöön hyödyntämällä leikillisyyttä robotteina. Tavoitteena on, että kuvasarja nostaisi lapsissa sisäistä motivaatiota sekä muistuttaisi lapsia omasta positiivisesta toiminnasta, mahdollisuuksista vaikuttaa siihen ja sen seurauksiin. Harjoitus tukee lasten loogista ajattelua ja tiedon käsittelyä vertaisryhmässä, ongelman ratkaisemista sekä ohjelmoinnin perusrakenteita kuvien ja ohjeiden lukemisen sekä kuvallisen ohjeistuksen rakentamisen kautta.

Lapsille siis kerrotaan aamukokoontumisessa tarina ”päiväkoti robotista”, joka on huomannut, että juuri tämä ryhmä kaipaisi toiminta avukseen robottikuvia. Mutta juuri tarvittavat kuvat ovat kadonneet metsään, koska vilkkaat ja vikkelät oravanpojat ovat käyneet viikonloppuna auki jääneen ikkunan kautta päiväkodilla leikkimässä ja samalla ne ovat ottaneet kuvat mukaansa. Kuvien löytymiseen tarvitaankin nyt lasten robottimaista apua. Lapset saavat tuokion alussa (retkipaikan reunalla) kuvallisen ohjeistuksen toimia tutussa metsäympäristössä. Alku ohjeeseen kuuluu aarrekartta, johon on piirretty geometrisia symboleja, joiden avulla on laskettu askelia, käännöksiä, kaatuneen puun ylitys sekä askeleet majaan.

Majaan löydettyään lapset löytävät sieltä pienen rullalla olevan kirjeen. Kirje sisältää geometrisiä symboleja, joita vain aikuinen osaa tulkita. Aikuinen saa kirjeessä ”ohjeen” näyttää lapsille kuvia iPadista. Näin jokainen pari näkee oman paikkakuvansa, mistä he lähtevät etsimään kadonneita kuvia. (Hyödynnetään lasten aikaisemmin iPadilla kuvaamia paikkoja, jotka tunnistetaan ja osataan etsiä.) Ohjekuvien paikoista löytyvät kuvat, jotka tuodaan ohjeen mukaan pussiin. Etsintätehtävän jälkeen jäämme vielä leikkimään metsään.

Seuraavana päivänä pienryhmässämme, tyhjennämme kuvapussin ja alamme pohtimaan mm. mitä kuvat esittävät. Useammasta kuvasta koostuva ohje kuvaa ulkovaatteiden pukemista. Nyt tehtävänä on järjestää kuvat oikeaan järjestykseen pohtien, mitä teemme ensin, sitten ja sen jälkeen. Kun kaikki ovat tyytyväisiä kuvalliseen sarjaan, joka ohjaa pukeutumista. Varmistamme, että käytännössä lapset myös osaavat tulkita kuvia ja ymmärtävän ohjeiden kaksisuuntaisuuden. Toiseen suuntaan puetaan ja toiseen riisutaan. Muistutuksena robottimaisuudesta,toiminnan

pilkkomisesta kuvin mietitään etteivät robotit vielä osaa puhua ja siksi tarvitsemme kuvia. Lopuksi kiinnitämme kuvasarjan eteiseen, jotta jokainen voi alkaa toimia ohjeen mukaan kuin robotti. Lasten toiveesta pukemisrobotti tarvitsee avukseen aikamittarin. Pohdinnan jälkeen löydämme naapuriryhmästä viiden minuutin tiimalasin. Se otetaan pukemisrobottien käyttöön.

Tehtävä tuki lapsia toimimaan vertaisryhmässään pareina. Lisäksi sekä ulkona että sisällä totutettu toiminta motivoi ja viritti lapsia juurikin tähän pukeutumistehtävään. Lasten innokkuus motivoi itse itsessään toimimaan ohjeiden mukaan ja kuvallisuus tuki ryhmän pienimpiäkin. Lapset heittäytyivät tehtävään ja "laskivat" tarkasti mukana symbolien mukanaan tuomia ohjeita. Kuvallinen itse löydetty ja rakennettu ohjesarja toimii sekä tukee käytännössä pukeutumistilannetta uudella innokkuudella.

Välineitä, joita käytimme ovat iPad, lasten aikaisemmin ottamat kuvat, piirretty aarrekartta, kirje, kangaspussi ja laminoituja kuvia, kaksi eriväristä nuolta kuvaamaan suuntaa ulkovaatteet päälle ja riisutaan sekä tiimalasi.

Ohjelmointia kehollisesti 4–6-vuotiaden integroidulle erityisryhmälle, varhaiskasvatus

Nimetön,

1.4.2022

Kehollisuus ja ohjelmointi / Varhaiskasvatus 

Kohderyhmä: Varhaiskasvastus 4-6-vuotiaat / Integroitu erityisryhmä

Kesto: Yksi yksittäinen tuokio, n. 30-45 min

Suunnitelma on tehty ajatellen omaa työpaikkaani ja ryhmää, jossa työskentelen. Kyseessä on integroitu erityisryhmä, jossa on tällä hetkellä 12 lasta. Iältään lapset ovat 4-6-vuotiaita ja osalla heistä on erityisen tuen tarpeita.

Suunnitelmaan on saatu ideoita ja inspiraatiota Kankaan ja Vartiaisen (2019) teoksesta ”Ohjelmoinnin ABC varhaiskasvatuksessa” (http://hdl.handle.net/10138/301730).

Aihepiirin valinta ja rajaus

Varhaiskasvatuksessa työtä ohjaavat Varhaiskasvatussuunnitelman perusteet (OPH 2018), jonka perusteella paikallisesti on laadittu omat Varhaiskasvatussuunnitelmat. Käytän suunnitelmassa pääosin suoria lainauksia valtakunnallisista vasu-perusteista, mutta olen poiminut lainaukset niin, että ne vastaisivat mahdollisimman hyvin paikallisen vasun painotuksia.

> *”Lapset tutustuvat matematiikkaan ja sen osa-alueisiin havainnollisen ja leikinomaisen toiminnan myötä. - - Lapsia innostetaan pohtimaan ja kuvailemaan matemaattisia havaintojaan ilmaisemalla ja tarkastelemalla niitä esimerkiksi kehollisesti tai eri välineiden ja kuvien avulla.” (OPH, 2018, s. 46.)*

Koska kyseessä on tuokio varhaiskasvatuksessa, luonnollinen tapa lähestyä aihetta on leikki ja leikinomaisuus. Kehollisuus näkyy vahvasti ryhmässämme muun muassa tukiviittomien käyttönä, joten oli myös luonnollista yhdistää toimintaan oma keho ja sillä toimiminen.

> *”Tavoite on, että lasten omakohtaisten kokemusten myötä herää ymmärrys siitä, että teknologia on ihmisen toiminnan aikaansaamaa.” (OPH, 2018, s. 47.)*

Paikallisessa varhaiskasvatussuunnitelmassa korostuu lapsen osallisuus ja tekemällä oppiminen, mikä myös tukee oman kehon käyttöä ohjelmointiin tutustumisessa ja algoritmisen ajattelun kehittämisessä. Tavoitteena on, että lapset huomaavat, miten suuri merkitys on täsmällisillä ohjeilla ja niiden noudattamisella, jotta päästään toivottuun lopputulokseen.

> *”Varhaiskasvatuksen tavoitteena on innostaa lapsia liikkumaan monipuolisesti sekä kokemaan liikunnan iloa. - - Varhaiskasvatuksen tehtävänä on kehittää lasten kehontuntemusta ja -hallintaa sekä motorisia perustaitoja, kuten tasapaino-, liikkumis- ja välineenkäsittelytaitoja.” (OPH, 2018, s. 47–48.)*

Tavoitteena on kehittää lapsen kehontuntemusta ja motoriikkaa – harjoituksessa käydään läpi eri kehonosia, nimetään kehonosia ja liikutaan eri tavoin. Tärkeä elementti on myös ohjeiden kuuntelu ja noudattaminen ja sitä kautta myös oman kehon pysäyttäminen/pitäminen paikallaan ohjeen mukaan.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Ohjelmointiin liittyen tavoitteena on saada lapset huomaamaan, mikä merkitys ohjeilla ja komennoilla on ja kuinka täsmällisiä ohjeita antaen ja noudattaen päästään haluttuun lopputulokseen. Kehollisesti tavoitteena on lisätä ymmärrystä siitä, että oma keho on juurikin jokaisen itsensä hallinnassa ja jokainen ihminen itsessään tekee päätöksen noudattaako ohjetta – sen sijaan kone toimii juuri niin kuin ihminen on käskenyt sen toimia.

Tarkoituksena ei ole arvioida lasten osaamistasoa, vaan pikemminkin sitä, miten opettajan toiminta ja tämä suunniteltu tuokio tukevat algoritmisen ajattelun ja kehotuntemuksen kehittymistä. Voidaan pohtia, onko toiminta vaativuudeltaan kohderyhmälle sopivaa ja miten lapset innostuvat toiminnasta sekä kiinnittyvätkö he siihen. Toki opettaja tekee myös havaintoja lapsista toiminnan aikana, jotta voi suunnitella seuraavan tuokion. Tuokion lopuksi lasten kanssa keskustellaan, mitä on tehty, mikä oli mukavaa ja opittiinko kenties jotakin uutta.

Työskentelyvälineet ja opetusmenetelmät

Tuokiota varten tarvitaan:

- tila, jossa on mahdollista liikkua
- CD-soitin tai muu median toistolaite
- noppa tai noppia
- tulostetut/piirretyt ohjekortit
    - liikkumistavat (käveleminen, hyppimien, kieriminen, ryömiminen, konttaaminen, kyykkykävely, karhukävely..)
    - suunnat esim. nuolina (eteen, taakse, oikealle, vasemmalle)
- merkkikartioita tai muita esineitä, joita voidaan käyttää merkitsemään kohdepaikkaa
- laatikko tai muu vastaava, johon voidaan kätkeä aarre

Tuokio aloitetaan yhteisellä laululeikillä, jossa toimitaan ohjeen mukaan. Tähän tuokioon valitsin lauluksi Siinan Taikaradio – Mun jalat kulkee näin (https://youtu.be/zfsx0nmX270). Kyseisessä laulussa liikutaan eri tavoin, kosketaan eri kehonosilla toisia kehonosia ja pysähdytään välissä, kun sanotaan ”stop”. Opettajana osallistun toimintaan ja näin omalla esimerkilläni innostan lapsia.

Seuraavaksi tarvitaan noppaa ja ohjekortteja sekä tilaa, jossa pääsee esteettä liikkumaan moneen suuntaan. Mikäli käytettävissä on iso taskullinen noppa, johon ohjekortit voi pujottaa, hyödynnetään sitä. Mikäli ohjekorteille ei ole noppia, voidaan kortit vain levittää kuvapuoli alaspäin lattialle, pöydälle tai johonkin sopivaan paikkaan. Maksimissaan käytössämme on siis kolme noppaa (tavallinen, liikkumistapojen ja suuntien noppa) ja vähintään yksi tavallinen noppa.

Leikin alussa kaikki leikkijät ovat tiheässä piirissä, jonka keskelle ensimmäisenä vuorossa oleva leikkijä heittää nopat. Saadaan näkyviin liikkumistapa, suunta ja liikuttava määrä – esimerkiksi kieriminen, taaksepäin ja 3. Nyt jokainen leikkijä saa tulkita tämän koodin ja lähteä sen mukaan kierimään. Yksi aikuisista voi toimia avustajana ja kuljettaa nopat aina seuraavan heittäjän luokse. Näin mennään, kunnes jokainen on vuorollaan saanut heittää ”koodin” ja mikäli leikkijöitä on vähän, voidaan mennä useampi heittokierros.

Kun kierros on täynnä, jaetaan jokaiselle merkkikartio, jonka he asettavat siihen paikkaan, jossa silloin ovat. Tämän jälkeen palataan lähtöpisteeseen ja keskustellaan, miksi päädyimme eri paikkoihin, vaikka jokainen noudatti samoja ohjeita.

Toisessa leikissä jakaannutaan pareiksi. Opettaja arvioi, voidaanko tehtävä suorittaa niin, että toinen pareista sulkee/peittää silmänsä ja on vain parinsa sanallisen ohjauksen varassa. Tarvittaessa tämä voidaan toteuttaa myös silmät auki. Nyt tehtävänä on ohjata vuorotellen oma pari sanallisia ohjeita antaen oman merkkikartion luokse. Kun päästään ensimmäiselle merkkikartiolle, vaihdetaan rooleja.

Kun kaikki parit ovat käyneet kummankin kartiolla, kokoonnutaan jälleen yhteen ja keskustellaan, mikä oli vaikeaa ja mikä helppoa sekä millaisia ohjeita kaverille annettiin.

Tuokion lopuksi lähdetään etsimään aarretta, jonka opettaja on kätkenyt ennen tuokion alkua. Tämä voi olla esimerkiksi tarroja laatikossa. Nyt opettaja ei ohjaakaan sanallisesti vaan ainoastaan näyttää ohjekorteilla ja nopalla ”koodin”, jonka mukaan liikutaan. Kun joku tai jotkut pääsevät aarteen luo, avataan aarre ja istuudutaan vielä miettimään, missä asioissa ohjeen noudattamisesta voi olla hyötyä tai missä kaikkialla vastaan tulee ohjeita/sääntöjä, joita on noudatettava. Tuokio päätetään ohjeistamalla lapset seuraavaan toimintaan.

Itse toiminta (liikkuminen ja leikkiminen) on lapsille riittävän mielekästä motivoimaan ja innostamaan toimintaan. Opettaja voi tarvittaessa kehittää taustalle tarinaa merirosvojen aarrejahdista. Omalla innostumisella ja eläytymisellä sekä lasten kehumisella ja kannustuksella uskon, että päästään toteuttamaan mukava tuokio.

Toiminnan voi myöhemmin yhdistää iPadilla tehtäviin harjoituksiin ja kokeilla, miten koodit toimivat esimerkiksi ScratchJr:ssa. Tällöin voidaan muistella, miten eri lähtöpaikka, koodien tarkka noudattaminen ja jokaisen oma toiminta vaikutti lopputulokseen.

Robottileikkejä 4–6-vuotiaille, varhaiskasvatus

Raisa Miettinen, CC BY-SA 4.0

1.10.2021

Tuntisuunnitelma 4-6- vuotiaille, robotit

1. Lapset saavat piirtää ja suunnitella oman robotin.

- Piirtäminen tapahtuu pienryhmissä, että kaikilla olisi työskentelyrauha.

    - Kierrän kyselemässä ja kirjaamassa, mitä heidän robottinsa tekisi ja miten se toimisi.

Tavoitteena: luoda kiinnostus aiheeseen, tukea luovuutta

Tarvikkeet: tusseja ja paperia

2. Kun kaikkien robotit ovat valmiit, lapset esittelevät robottinsa muulle ryhmälle ja kertovat niistä.

-Mietimme myös missä oikeasti on robotteja/automatiikkaa

-Juttelemme, että roboteilla ja koneilla on oma kieli (vähän kuin salakieli/salakoodi), jolla toimintakäskyt annetaan. Juttelemme myös siitä, että käskyjen tulee olla yksinkertaisia, jotta kone ne ymmärtää.

-Lopuksi jokainen lapsi saa kertoa, tai näyttää, muille yhden yksinkertaisen komennon (esimerkiksi liike, ilme, taputus)

Tavoite: antaa rohkeutta kertoa omista ideoistaan, saada jonkinlainen käsitys siitä että ohjelmointi tapahtuu yksinkertaisilla käskyillä, omalla kielellään

Tarvikkeet: ei tarvitse välineitä

3. Jatkamme toimintaa myöhemmin metsäretkellä.

- Lapset saavat etsiä, mitä metsästä löytyviä asioita voisi käyttää robottien kielen symboleina?

- Miten robotti sanoisi “hei”, montako asiaa pitäisi löytää?
- Entä jos robotti sanoisi pitkän asian tai lyhyen asian? Löytyykö pitkä/lyhyt luonnon esine?
- Entä jos robotti haluaisi vuorotella pitkä, lyhyt, pitkä, lyhyt? Osaatko tehdä sarjan?
- Miltä sarja näyttäisi liikkeeksi muunnettuna?
- Lapset voivat keksiä myös muita sarjoja ja miettiä(arvioida) pystyykö niitä toteuttamaan löytyvillä materiaaleilla tai muilla tavoin?

Tavoite: harjoitella monilukutaitoa, loogista päättelyä, soveltamista ja sarjoittamista

Tarvikkeet: metsästä löytyviä luonnon esineitä

4. Toiminta jatkuu pareittain. Muut saavat sillä aikaa leikkiä vapaasti. Pari tulee pressulle, johon olen teipannut 4x4 ruudukon, jonka molempiin reunoihin olen merkannut myös lähtö- ja maaliruudun. (Samankaltainen tehtävä oli nimellä viruskaappari opinsys materiaaleissa

https://opinsys.fi/wp-content/uploads/2017/05/viruskaappari_260517_pien-1.pdf)

Toinen lapsista asettaa aiemmasta tehtävästä tutun robotin koodin palasen johonkin ruudukon ruutuun. Sitten hän ohjeistaa parinsa ruutu kerrallaan ruudukon lähtöpisteestä hakemaan puuttuvan palan ja sieltä maaliin. Ohjeistavan lapsen kannattaa olla maalin puolella, jotta ohjeistettava näkee hänet.

-Vaihdetaan vuoroja

-Jos tehtävä on liian helppo voi tehtävää vaikeuttaa niin, että ohjeita antava on ohjeistettavan takana. Tai niin että pitää palata myös takaisin.

Tavoite: harjoitella yhteistyötä, loogista ajattelua, ohjeiden antamista ja niiden mukaan toimimista, suuntakäsitteitä (oikea, vasen, tänne, tuonne, eteen, taakse jne.)

Tarvikkeet: Pressu, johon on teipattu 4x4 ruudukko

-Kaiken toiminnan aikana havainnoin lapsia suhteessa tehtäviin: onko liian helppoa/liian vaikeaa, mitä kannattaisi jatkossa tehdä.

- Havainnot ryhmästä vaikuttavat myös siihen, monellekko päivälle toiminta jakautuu. Toiminta on helppo jakaa useamman päivän ajalle ryhmästä riippuen.

-Jos lapset innostuvat paljon, voimme esim. myöhemmin askarrella heidän suunnittelemansa robotit, tai ottaa ruudukkopressun käyttöön myös toisella retkellä. Ruudukkoon saisi lisää haastetta myös suurentamalla sitä tai lisäämällä osiin ruutuihin “esteitä”

Lähteet:
https://opinsys.fi/wp-content/uploads/2017/05/viruskaappari_260517_pien-1.pdf

Ohjelmointia liikunnallisin ja leikillisin keinoin 5-6-vuotiaille, varhaiskasvatus

Nina Parviainen, CC BY-SA 4.0

6.2.2021

Aihepiirin valinta ja rajaus

Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Sovellan ohjelmointia liikuntatuokion tehtäviin. Tuokio on suunnattu 5-6 -vuotiaille lapsille. Heille leikillisyys ja liikunnan ilo sekä toimiminen yhdessä vertaisryhmän kanssa on ominainen tapa oppia.

Oppimiskokonaisuuden tehtävät ovat leikillisesti ohjelmoinnilliseen ajatteluun johdattelevia ja tukevat lapsen algoritmisen ajattelunkehittymistä. Yhdistämme myös Bee-botrobotin käyttöä liikuntatehtäviin. Erilaiset laitteet ja ohjelmat kuuluvat nykypäivänä jo pientenkin lasten elämään. Ne ovat mukana lasten arjessa, niiden kanssa toimitaan harjoitellen kieltä ja arjen taitoja. Oppimiskokonaisuudessa harjoitellaan myös ryhmä- ja parityöskentelytaitoja, oman vuoron odotamista, käytetään mielikuvitusta, kerrataan muotojen nimeämistä, lukujen luettelemista ja lukumerkkejä.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

Ohjelmoinnillisesti harjoitellaan tehtävän purkamista osiin, ohjeiden antamista sanallisesti ja symbolien avulla, koodikieltä ja ohjelmoidaan yksinkertaista laitetta (Bee-bot). Liikunnallisesti ja leikillisesti harjoitellaan kehon liikkuvuutta, tuetaan perusliikuntataitoja ja koetaan yhdessä toimiessa liikunnan iloa.

Miten opettaja tai joku muu arvioi opiskelijan osaamista? Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Opettaja ja lastenhoitaja havainnoivat lasten toimintaa esim. matemaattiset valmiudet, karkeamotoriset taidot, ryhmätyöskentely, ohjeiden kuuntelu ja ohjeiden mukaisesti toimiminen, kielelliset valmiudet. Havainnointeja kirjataan ja niitä voidaan hyödyntää lasten varhaiskasvatussuunnitelmia/esiopetussuunnitelmia laatiessa ja arvioitaessa.

Toimintaa valokuvataan myöhemmin tapahtuvaa portfoliotyöskentelyä varten.

Miten oppilas arvioi omaa osaamistaan?

Lasten kanssa keskustellaan tuokioiden lopuksi. Lasten ajatuksia kirjataan ylös ja niitä voidaan liittää myöhemmin yhdessä lasten kanssa ryhmän/lasten portfolioihin tuokioilta otettujen valokuvien lisäksi.

Työskentelyvälineet ja opetusmenetelmät

Oppimiskokonaisuus toteutetaan päiväkodin liikuntatilassa tai muussa isohkossa vapaassa tilassa. Tuokiolle osallistuu kerrallaan 20 lapsen ryhmästä puolet. Toiminnan toteutukseen osallistuu opettajan lisäksi ryhmän lastenhoitaja. Lapsiryhmä on jo aiemmin työskennellyt yhteistoiminnallisissa 3-4 lapsen pienryhmissä. Myös parityöskentelyä on käytetty aikaisemmin. Lapset ovat jo aiemmin

tutustuneet Bee-botrobotin toimintaan ja harjoitelleet sen ohjelmointia tiettyyn paikkaan ruudukolla.

Mitä välineitä oppikokonaisuudessa käytetään?

Oppikokonaisuuden tehtäviin liittyy vahvasti visuaalinen tuki. Kuvatuen ja tehtäväkorttien valmisteluun kannattaa varata tarpeeksi aikaa jos päiväkodilla ei ole vielä valmista kuvamateriaalia/liikuntatehtäväkuvia valmiina. Kuvatuen tekemisessä voi hyödyntää esim. Papunetin kuvatyökalua:

https://kuvatyokalu.papunet.net/#/muokkaa/1961413

Tarvittavat välineet: leikkirobotti/robottikuva, kuvakortit numeroista 1, 2, 3, kuvakortit liikuntatehtävistä, kuvat/lattiamuodot muodoista: ympyrä, neliö, kolmio, suorakulmio, Bee-bot robotteja ja taskullisia ruudukoita Bee-boteille, Bee-botruudukon taskuihin sopivat liikuntatehtäväkortit.

Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Kokoonnutaan aluksi keskustelemaan yhdessä roboteista. Tässä mukana on leikkirobotti tai iso kuva leikkirobotista. Liikuntatehtäviin annetaan ohjeistus sanallisesti. Sanallisia ohjeita tukee esillä oleva kuvamateriaali. Kuvatuki auttaa tehtävän muistamista ja pilkkomista osiin.

Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Leikillisyys, liikunnallisuus, visuaalisuus ja yhteinen keskustelu virittävät lapsia aiheeseen.

Miten opetuksessa rohkaistaan yhteistyötä?

Tuokion tehtäviin liittyy sekä pari että ryhmätyöskentelyä. Johdattelua aiheeseen ja arviointia toteutetaan yhdessä keskustellen.

Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Päiväkodissa moniin arjen toimintoihin (esim. pukeminen, käsienpesu, ruokailu) liittyy samana toistuvia tehtäväohjeita. Toisen tuokion Bee-botruudukkotehtävään edellytetään aikaisempaa tutustumista Bee-botrobotin toimintaan ja käyttöön.

Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelua?

Kiinnitetään huomiota ja keskustellaan lasten kanssa ympäristössä esiintyviin laitteisiin ja ohjelmiin; päiväkodissa esimerkiksi automaattisesti syttyviin ja sammuviin valoihin, automaattisesti toimiviin vesihanoihin. Arjen toimintoihin voi myös liittää kuvatauluja tehtäväjärjestyksestä ja pilkkoa toimintaa yksittäisiin ohjeisiin.

ROBOTTIJUMPPA

1. TUNTISUUNNITELMA

1. Kokoonnutaan istumaan salin lattialle keskiympyrään. Kerrotaan lapsille, että tänään jumppaamme kuin robotit. Kysellään**, mitä lapset tietävät roboteista, mitä robotit osaavat tehdä, miten ne**

liikkuvat, millaista ääntä roboteista lähtee ym. Varsinkin visuaalisuutta puheen tueksi tarvitseville lapsille **mukana voi olla leikkirobotti tai iso kuva robotista**. Lasten ajatuksia kirjataan ylös.

2. Alkulämmittelynä noustaan ringissä seisomaan ja lasketaan 1-2-3... Seuraavalla kierroksella sovitaan, että numeroa 2 ei sanota vaan mennään kyykkyyn ja noustaan ylös (yksi, kyykkyyn-ylös, kolme) Seuraavaksi sovitaan että 3 on hyppy (yksi, kyykkyyn-ylös, hyppy). Sitten sovitaan että 1 on pyörähdys paikalla. Tehdään kierros ilman puhetta (pyörähdys paikalla, kyykkyyn-ylös, hyppy).

   Tehtävää havainnollistetaan kuvatuella: numerot 1, 2, 3 ja tehtäväkuvat: pyöriä, kyykky, hypätä.

   (Tehtävä löytyy materiaalista: Ohjelmoinnin ABC varhaiskasvatukseen).

3. Tässä leikissä robotit liikkuvat symbolien (kuvat muodoista tai valmiita lattiamuotopaloja) avulla. Nämä robotit ymmärtävät vain muotokieltä. Robottikielessä **neliö tarkoittaa askelta eteen päin**, **ympyrä tarkoittaa että pyöritään paikalla ympäri kaksi kertaa, kolmio tasahyppyä eteen päin ja suorakulmio askelta taakse päin.** Opettaja näyttää lapsille ensin yhtä muotoa kerrallaan. Kun lapset muistavat ohjeet, voidaan näyttää kahta muotoa kerrallaan. Lopuksi lapset voivat suunnitella robottiviestin laittamalla muodot lattialle/seinälle haluamaansa järjestykseen. Sitten painetaan robotin virtanappia ja tehdään muotokielellä suunnitellut liikkeet.

   (Tehtävä löytyy materiaalista: Ohjelmoinnin ABC varhaiskasvatukseen).

4. Robotit ovat saaneet niin monta ohjetta, että menevät aivan sekaisin. Konemaisen "kulmikkaan" musiikin rytmissä liikutaan salissa kulmikkain liikkein (lapset kuvittelevat millaisia ovat robottiliikkeet ja liikkuvat robotteina). Kun musiikki loppuu, robottien virta on lopussa ja ne lysähtävät salin lattialle.

5. Laitetaan rentouttavaa kepeää sanatonta musiikkia soimaan. Robotit latautuvat (lapset rentoutuvat salin lattialla) opettajan ja lastenhoitajan heilutellessa huiveja kunkin lapsen luona vuorotellen.

6. Lapset voivat nousta istumaa. Keskustellaan lopuksi, mitä lapset tykkäsivät robottijumpasta? Mikä oli helppoa? Mikä oli vaikeaa kun jumppasi robottina? Mitä voisi vielä harjoitella lisää? (Opettaja tai lastenhoitaja kirjaa lasten ajatuksia).

2. TUNTISUUNNITELMA

1. **Leikitään robotteja ja sokkoa.** Kysytään kuka lapsista on vapaaehtoinen sokoksi silmät peitettynä huivilla. Muut ovat robotteja ja liikkuvat sovitulla alueella robottimaisin liikkein äännellen kuin robotit. Sokko yrittää saada robotin kiinni äänen perusteella. Jos sokko saa robotin kiinni, robotista tulee uusi sokko**.**

2. **Leikitään robottia ja ohjaajaa.** (Robotin ohjaaminen sanallisin ohjein parityöskentelynä) Lapset ovat robotteja ja opettaja toimii ensin robottien ohjaajana. Robotit ovat oppineet vähän puhetta ja ymmärtävät, mitä tehdä kun ohjaaja antaa sanallisen ohjeen esim. ota askel eteenpäin, ota askel

vasemmalle, ota askel taaksepäin. Kun lapset ovat oivaltaneet idean ja osaavat liikkua ohje kerrallaan, muodostetaan parit (robotti ja ohjaaja). Robotin silmät sidotaan. Sovitaan yhdessä mihin salissa robotit ohjataan. Ohjaaja käynnistää robotin painamalla ”virtanappia” ja ohjaa sanallisesti robotin yksi ohje kerrallaan sovittuun paikkaan. Vaihdetaan vielä rooleja.

(Tehtävää muokattu materiaalista Ohjelmoinnin ABC varhaiskasvatukseen).

3. **Ohjelmoidaan Bee-botia ruudukolla, jossa on kuvia liikuntatehtävistä.** Lapset jaetaan yhteistoiminnallisiin pienryhmiin, jotka ovat heille jo entuudestaan tuttuja tässä ryhmässä. Tehdään ryhmässä Bee-bottia ohjelmoiden liikuntakortin tehtävä, johon Bee-bot on ohjelmoitu. Lapset ohjelmoivat Bee-bottia vuorotellen.

4. Kokoonnutaan lopuksi ja keskustellaan yhdessä miltä jumppa tuntui, tuntuiko jokin vaikealta/helppolta? Miltä tuntuu jos ei näe mitään? Miten osaa toimia jos ei näe mitään? Miten roboteille voi antaa ohjeita? Mitä tapahtuu jos robotti ei ymmärrä ohjetta? Mitä lapset toivoisivat vielä tehtävän uudestaan? (Opettaja tai lastenhoitaja kirjaa lasten ajatuksia).

Ohjelmoinnillisen ajattelun harjoittelua musiikin ja liikunnan avulla, varhaiskasvatus

Johanna Tuupainen,

14.3.2021

Tuntisuunnitelma varhaiskasvatukseen

Suunnittelin tuokiot varhaiskasvatukseen, ryhmään jossa on yhdeksän 2-5 vuotiasta lasta. Tuokiot eivät siis vastaa pituudeltaan kahta oppituntia, tarkoitus olisi toteuttaa tuokiot omassa ryhmässäni niin, että kaikki lapset voisivat osallistua ja jaksaisivat olla mukana, tietysti aina vähän ikätason huomioiden.

Robotit ovat olleet ryhmässämme esillä viime aikoina, ja robotti teemaan liittyvät myös nämä tuokiot. Lapset askartelivat muutamia viikkoja sitten omia robottejaan ja ideoivat mitä oma robotti tekee. Tämä ja aiheen jatkunut ylläpito toimii motivaationa suunnittelemilleni hetkille. Orientaationa aiheeseen voisi siis pitää erillisen askartelutuokion, jossa jokainen saisi tehdä oman robottinsa tai vaikka värittää valmiin värityskuvan ja miettiä mihin tarkoitukseen oma robotti on suunniteltu.

Tuokioiden sisällöiksi valikoitui nyt musiikki ja liikunta. Molemmissa tuokioissa toiminnallisuus ja elämyksellisyys ovat isossa roolissa ja koska lasten ikäjakauma on kuitenkin melko suuri, on suunnitelussa pyritty ottamaan huomioon, että jokainen lapsi voi osallistua jollakin tavalla. Myös tavoitteet ovat osin yhtenevät molemmilla tuokioilla. Tarkoitus on herätellä ohjelmoinnillista ajattelua harjoittelemalla täsmällisten, yksinkertaisten ohjeiden antamista ja toteuttamista, symbolin ja toiminnan yhteyden ymmärtäminen (eli koodikielen hahmottaminen), rytmiikka, värit ja muodot sekä kasvattajan aiheen kielellistämisen harjoitteleminen. Tämän ikäisillä varsinaisen arvioinnin sijaan pyrkisin havainnoimaan osallistumisen tasoa ja innokkuutta sekä arvioisin enemmänkin tavoitteiden toteutumista kunkin lapsen kohdalla. Avuksi havainnointiin pyytäisin toisen kasvattajan, joka voisi keskittyä pelkästään tähän.

MUSIIKKI

Tavoitteet, tarvikkeet ja ennakkovalmistelut:

”Yhteisten” tavoitteiden lisäksi mm. ohjeiden kuunteleminen ja rytmin/ohjeen mukaan soittaminen. Esivalmisteluina ”mä olen robotti”- kappaleen etsiminen valmiiksi, mikäli käytettävissä on valoprojektori tms. tehoste, voi sen ottaa tunnelmaa luomaan, ”koodikielen” symboleita esim. 5 kpl kutakin. Klaveesit tai muut sopivat soittimet rytmiharjoituksiin. Voi toki taputtaa vaikka käsin. Norsun ja apinan jälkiä esim paperille piirrettynä (erikseen toisistaan, norsun jälkeä piirretään arkille yksi suuri, apinan jälkiä kaksi pienempää).

Toteutus:

Tuokio alkaa kuuntelemalla Siinan taikastudion kappale ”Mä olen robotti”. Laitetaan mahdollisesti valoprojektori luomaan tunnelmaa ja lapset saavat liikkua tilassa kuka mitenkin robottia jäljitellen.

Istutaan alas ja perehdytään Läps- Taps kieleen. (Ohjeet löytyivät ohjelmoinnin ABC-oppaasta).

Neliö= lyö kädet kerran yhteen

ympyrä= tömistä jaloilla lattiaan

kolmio= taputa käsiä nopeasti

tähti= nosta kädet ylös

Symboleita on hyvä valmistaa useita samoja, jolloin saadaan pidempiä koodisarjoja, joita sitten lasten kanssa mietitään ja joiden mukaan toimitaan. Lopuksi voidaan miettiä vielä yhdessä muutama merkki & toiminta lisää.

Jatketaan rytmittelyä ja leikillä, jonka ohjeet löytyivät **”matematiikkaa kehollisesti ja liikkuen”- kurssin materiaaleista**. Tässä hieman muokattuna.

Lauletaan ”piiri pieni pyörii, lapset siinä hyörii

norsu tallaa lattiaa, apinakin avustaa”

Mietitään millainen eläin norsu on ja vaikka aiheeseen sitoen, että jos siitä tekisi jonkin koneen tai laitteen niin miltähän se mahtaisi näyttää tai kuulostaa, ehkä liikkua. Matkitaan norsun raskasta liikkumista. Samoin apinan kohdalla. Jokainen saa omat klaveesit ja soitetaan rytmisarjoja opettajan lattialle laittamien jälkien mukaisesti: norsun jälki on yksi hidas, apinan kaksi nopeaa. Voidaan myös liikkua samanlaisia sarjoja jälkien paikkoja muutellen tai niin että jokainen saa tehdä vuorollaan oman sarjan.

Lopuksi runoillaan ”Mikä robotti” runo:

Robotit eivät ole vain peltiä ja muttereita, ne ovat myös aika taitureita.

Ne osaavat säätää autoja ja pelata palloa ja nurmikolla myyränkekoja talloa.

Niillä ei ole tunteita, mutta ne osaavat niitä lukea, ja ihmisen avulla tunteet sanoiksi pukea.

Sensoreilla robotti löytää ja havainnoi ja kaiken tärkeän raportoi.

Sydäntä ei ole, mutta tietokone kyllä, näin robotti pitää tehokasta toimintaa yllä.

(Fretland van Voorst, J. & Morgan, P. 2018, 4.)

LIIKUNTA

Tavoitteet, tarvikkeet ja ennakkovalmistelut:

”Yhteisten” tavoitteiden lisäksi tavoitteena mm. parityöskentelyn harjoittelu, oman vuoron odottaminen, ruudukkotehtävässä tavoitteena harjoitella jo hieman loogisuutta kun yritetään saada pelaaja pysymään tai tippumaan ruudukolta.

Toteutus:

Alkulämmittelyyn tarvitaan iso pehmonoppa, jota jokainen lapsista saa vuorollaan heittää. Ennen nopan heittämistä lapsi saa keksiä liikkeen, jota sitten toistetaan nopan silmäluvun mukainen määrä.

Seuraavana tutustumme Nestori neliökieleen (ohjelmoinnin abc varhaiskasvatukseen).

Neliö = kävellä

ympyrä = tasajalkahyppy

kolmio = osoittaa suuntaa

tähti = vilkuta

symboleista muodostetaan jälleen liikesarjoja, joita yhdessä toistetaan ja muokataan erilaisiksi. Lapset saavat myös käydä pareittain suunnittelemassa liikesarjan, jonka toiset toistavat.

Jokainen pääsee myös omalla vuorollaan kuuntelemaan ohjeita ja liikkumaan niiden mukaisesti. Suurelle paperille (meillä kirjapainosta saadusta paperirullasta useita siivuja teipattu toisiinsa) piirretään ruudukko, jossa olisi tarkoitus mahtua liikkumaan (esimerkiksi 7X7 ruutua?). Yksi lapsista asettuu johonkin kohtaan ruudukkoa ja muille lapsille jaetaan jokaiselle yksi lappu, jossa on nuolisymboli (oikealle, vasemmalle, eteen, taakse). Vuorotellen jokainen antaa ruudukolla olijalle oman liikeohjeensa ja jännityksellä seurataan, pysyykö liikkuja ruudukolla vai tipahtaako

sieltä pois. Peliä voidaan pelata niin, että joko ruudukolla olija on tarkoitus saada pysymään siinä tai yritetään saada ulos ruudukolta. Kumminkin päin on mahdollisuus jo hieman harjoitella taktikointia sillä, kenen hallussa oleva nuoli kannattaa missäkin kohti käyttää ja näin harjoitella myös yhteistyötä. Nuoliohjeita on hyvä olla reilummin ja jokainen voi nostaa omansa vaikka hatusta. Mikäli tuntuu, että ruudulla olijan tiputtaminen voi tuntua ikävältä, voidaan vaikka sopia että ruudukolla olija on robotti, joka ohjeiden mukaan liikkuu.

Lopuksi muututaan vielä kaikki roboteiksi, jotka liikkuvat tilassa/ suorittavat omaa tehtäväänsä tms. niin kauan kun musiikki soi. Kun musiikki lakkaa, loppuu myös roboteista virta siihen paikkaan. Kun musiikki jatkuu, jatkuu myös liike.

LÄHTEET:

Fretland van Voorst, J. & Morgan, P. 2018. Mitä ihmettä! Robotit. Lasten keskus

Ohjelmoinnin ABC- varhaiskasvatuksessa.

http:/hdl.handle.net/10138/301730

Siinan taikastudio: mä olen robotti.

https://youtu.be/78sCS44TyIk

Mikä tekee koirasta koiran? - harjoituksen sovellus, varhaiskasvatus

Nimetön,

11.3.2021

Sovelletaan Koodauksen abc:stä tuttua Mikä tekee koirasta koiran? - harjoitusta.

Tuntiaihe valikoitui lasten kiinnostuksen kohteen ja vahvuuksien mukaan. Yksi ryhmän lapsista on saanut perheeseensä koiran ja puhuu paljon koirasta. Lisäksi pienryhmässä on muutama lapsi, ketä kiinnostaa tällä hetkellä muovailu tosi paljon. Pienryhmässä on läsnä 4 lasta, ikähaitarilla 2,5-3 v.

Tavoitteena toiminnalle on viettää yhdessä tämän pienryhmän kanssa kiireetöntä ja keskeytyksetöntä aikaa muovailun parissa: harjoitella silmän ja käden yhteistyötä, harjoitella ja vahvistaa suomen kieltä (2 lapsen äidinkieli on joku muu kuin suomi), tukea lasten keskinäisten suhteiden syntymistä (lapset toisilleen vielä suht uusia tuttavuuksia) ja vuorovaikutus- ja keskustelutaitoja (kuuntelen, odotan omaa puheenvuoroani, rohkenen kertoa omia ajatuksiani). Lisäksi harjoittelemme pienille tarpeellisia ensimmäisiä ohjelmoinnin valmiuksia: käsite ja määritelmä koiralle (mitkä asiat tekee koiran, mikä erottaa koiran kissasta...), havainnoidaan ja sanoitetaan yksityiskohtia, vaihe-vaiheelta -ajattelu).

Vaikka keskustelu ja ilmiön tarkastelu tässä suunnitteluvaiheessa keskittyy koiraan, voisin veikata, että todennäköisesti keskustelu ja oppiminen laajenee myös luokitteluun eläimet tai esim. neljällä raajalla liikkuvat...

No, näin pienten lasten kanssa arviointi kohdistuu mun toimintaan kasvattajana: siihen kuinka pystyn toiminnan hetkellä muokkaamaan omaa toimintaani vastaamaan ohjattavissa olevan pienryhmän tarpeita ja suhteessa lasten kysymyksiin, keskittymiseen, sen hetkiseen vireystilaan yms. Arvioinnin pohjana on lasten sitoutuneisuus toimintaan, motivaatio ja jaksaminen sekä innostuneisuus ja tekemisen ilo, mikä kyllä näkyy koko kehossa.

Välineet: selkeäitä kuvia muutamasta eri rotuisesta koirasta (oikeita koiran kuvia netistä, A4 tai A3, tai joku hyvä koirakirja), muovisia koiraleluja (mahd. isoja mitä löytyy), muovailuvahaa, muovailuvälineinä puikkoja+ kaulin yms., Tabletti/ känny yms. millä ottaa kuva,

Jokainen saa ottaa muovailuvahaa ja lähdetään yhdessä työstämään muovailuvahasta yhteistä koiraa. Tässä kohtaa sovelletaan Mikä tekee koirasta koiran? -harjoituksen ohjetta, koska on pienistä lapsista kyse. Kasvattaja kannattelee "koiran kokonaisuutta" pyytäen lapsia tekemään koiralle korvan ja toisenkin, hännän jne. Nämä kaikki liitetään yhteen yhteiseen koiraan. Ajatus on, että kasvattaja vie keskustelua eteenpäin ja kiinnittää lasten huomion koiran kehonosien tarkasteluun avoimilla kysymyksillä tai sanoittaen/ nimeämällä koiran toimintaa sekä kehonosia. Samalla koko ajan nähtävillä koirien kuvat ja muoviset koiralelut, joita tarkastellaan ja tutkitaan. Lasketaan monta korvaa, raajaa, tassua, silmää... koiralla on. Välillä voidaan heittäytyä ja lähteä konttaamaan koiramaisesti tai läähättää kieli pitkällä... leikinomaisuus mukana ihan riippuen, miten lapset ottaa/ tarvitsee leikillisyyden ja kehollisuuden osana toiminnallisuutta.

Yhteistyöhön kannustan niin, että kuuntelemme toista, jokainen saa puheenvuoron, emme puhu toisen päälle, ojennatko kaverille..., autatko kaveria..., katso miten hienon korvan kaveri on tehnyt... jne.

Opittua asiaa voidaan tuoda esille esim. kun vertaillaan mitä tahansa esineitä, ilmiöitä... eli kun eri koirankuvia vertaillaan ja löydetään samankaltaisuuksia niin huomataan, että kaikilla koirilla (yleensä) on 4 raajaa, 2 silmää, 2 korvaa... tätä voidaan tuoda esille, kun määritellään esim. autoja autoleikissä (yksi suuri mielenkiinnon kohde myös ryhmässämme). Henkilöauto, kuorma-auto, mopo... kaikkia näitä käsitteitä määritellään eri sisällön avulla ja niiden määrittely vaatii tarkkaavaisuutta, havaintojen tekemistä ja yksityiskohtien huomaamista.

Pienillä lapsilla tämä suunniteltu tuntisuunnitelma ei todennäköisesti kestä 45min. vaan lyhyemmän ajan, mutta kun se toistetaan muutaman kerran 1-2 viikon aikana saadaan toiston, kertauksen ja toiminnallisuuden sekä leikillisyyden avulla luotua opintokokonaisuus. Usein on niin, että ns. pöydän ääressä tehdyt toiminnot "hyppäävät" mukaan ulkoleikkeihin, kuten esim. tässä tapauksessa pienryhmämme innostui pihalla leikkimään koiria. Tai että ruokapöydässä tai kävelyretkellä lapset palaavat aiheeseen jonkun kysymyksen toimiessa "keskustelunavaajana" esim. "mikä oli se koiran heiluva? (=tarkoittaen häntää). Tai että ajatukset jäävät "muhimaan" ja niihin on hyvä palata määritellessä jotain muuta käsitettä.

Kasvattajan mallintama ja sanoittama vaiheittainen ajattelu on hyvä pohja ohjelmoinnissa tarvittavalle ongelmanratkaisutaidoille ja erilaisten osien yhteensovittamiselle. On joku ongelma, pilkotaan se pienempii osiin ja lähdetään osa kerrallaan ratkaisemaan pulmaa, ja näin edetään osa ja ratkaisu kerrallaan kunnes kokonaisuus on selvillä.

Kolmiosainen ohjelmointituokioiden kokonaisuus, varhaiskasvatus

12.3.2022

Tuokio koostuu kolmesta erilaisesta tehtävästä, jotka toteutetaan eri päivinä. Ensimmäinen tehtävä toimii motivaattorina, koska lapset nauttivat siitä, että saavat ohjeistaa ope-robottia. Lapsille kerrotaan mitä ohjelmointi tarkoittaa, mihin sitä tarvitaan ja että harjoitusten avulla tutustumme ohjelmointiin ja algoritmiseen ajatteluun.

Tuokion tavoitteena on saada kokemuksia ohjelmoinnista, ymmärtää mitä ohjelmointi ja algoritminen ajattelu tarkoittavat, oppia sanallistamaan ohjeita tarkasti, harjoitella sijaintikäsitteitä, kehittää matemaattis-loogista päättelyä, oppia havainnollistamaan yksityiskohtia, harjoitella esineen määritelmien kertomista, saada kokemuksia ohjelmoinnin ja laulun yhdistämisestä sekä kokea iloa, onnistumista sekä yhteenkuuluvuutta ryhmän kanssa. (Kangas & Vartiainen, 2019.)

Arviointi: Opettaja arvioi tavoitteiden saavuttamista jatkuvasti ja antaa rakentavaa ja kannustavaa palautetta. Opettaja antaa palautetta suullisesti sekä peukkua näyttäen.

Tavoitteet ja niiden arviointi:

Vahvistaa ymmärrystä siitä, mitä ohjelmointi ja algoritminen ajattelu tarkoittavat -->Ymmärtävätkö oppilaat mitä ohjelmointi ja algoritminen ajattelu tarkoittavat?

Oppia sanallistamaan ohjeita tarkasti --> Onnistuiko lapsilta ohjeiden antaminen?

Harjoitella sijaintikäsitteitä -->Osasivatko oppilaat käyttää oikeita käsitteitä?

Oppia havainnollistamaan yksityiskohtia --> Onnistuiko yksityiskohtien havainnollistaminen?

Harjoitella esineen määritelmien kertomista --> Onnistuivatko kertomaan esineen määritelmiä?

Saada kokemuksia ohjelmoinnin ja laulun yhdistämisestä --> Onnistuiko laulun ja käsiliikkeiden yhdistäminen?

Kokea iloa, onnistumista sekä yhteenkuuluvuutta ryhmän kanssa --> Oliko tuokio mukava, välittyikö yhteenkuuluvuuden tunne?

Viskarit itsearvioivat osaamistaan ja omaa työskentelyään tuokion päätteeksi ympyröimällä hymynaamoja:

Tiedän, mitä ohjelmointi tarkoittaa: 😊:I ☹

Tein parhaani Harri-robotti -tehtävässä: 😊:I ☹

Tein parhaani Mikä tekee koirasta koiran? -harjoituksessa: 😊:I ☹

Tein parhaani Laululeikkitehtävässä: 😊:I ☹

Harri-robotti: (30min) ”Harjoituksessa opettaja esittää robottia, joka tottelee lapsia täsmällisesti” (Kangas & Vartiainen, 2019). Lapset yrittävät saada opettajan kokoamaan palikoista ensin tornin, sitten linnan. ”Opettajan tehtävä on toimia tarkalleen lasten ohjeiden mukaisesti. Lapset oppivat pian, että komentojen on oltava hyvin tarkkoja, tai robotti toimii väärin. Opettaja asettaa tarvittavat puupalikat pöydälle. Opettaja seisoo

paikoillaan tönkkönä ja kertoo lapsille esittävänsä robottia. Opettaja pyytää lapsia ohjaamaan Harria yksiselitteisillä komennoilla, kuten ”siirrä kättäsi vasemmalle” tai ”valitse kätesi vieressä oleva palikka”. Jos lapsi yrittää liian monimutkaistakomentoa tai komentoa, joka ei sisälly robotin sanavarastoon, opettaja pudistelee päätänsä.” (Kangas & Vartiainen, 2019.) Tehtävä on valmis, kun torni/ linna on rakennettu. ”Opettaja keskustelee lasten kanssa siitä, mihin komentoihin robotin voi olettaa voivan vastata ja mitä robottia todennäköisesti ei voi pyytää tekemään.” (Kangas & Vartiainen, 2019.) Vaihdetaan osia: opettaja ohjeistaa seuraavaksi lapsi-robotteja samaisessa tehtävässä.

Tehtävän tavoitteet: ”Harjoitellaan ongelman pilkkomista osiin, ohjeen täsmällistä sanallistamista, sijaintikäsitteitä ja matemaattis-loogista ajattelua” (Kangas & Vartiainen, 2019.)

Mikä tekee koirasta koiran? -tehtävä (30min): Käytetään valmiiksi opettajan valikoimia Alias-sanaselityskortteja. Lapsi kääntää kortin ja alkaa piirtää kuvassa olevaa esinettä. ”Muut yrittävät arvata, mikä se on. Kun joku arvaa oikein, pysähdytään miettimään, mikä oli se asia, joka teki piirroksesta sen mitä piirrettiin. Mietitään myös voisiko se kuitenkin olla myös jokin muukin asia.” (Kangas & Vartiainen, 2019.) Vaihdetaan piirtäjää.

Ohjelmoinnin ydintä on pystyä määrittelemään yksinkertaisesti käsitteitä ja tässä tehtävässä pyritään juuri siihen. Tehtävän tavoitteet: ”Yksityiskohtien havainnointi, ilmiön tai esineen määritteleminen, mielikuvan kommunikoiminen, vaiheittainen ajattelu” (Kangas & Vartianen, 2019) ilo ja yhteenkuuluvuus.

Laululeikki ”Kas metsämökin ikkuna”: (15min) ”Tuttu laulu, jossa tutut käsillä tehdyt viittomat rakentavat lauletun musiikin vierelle omaa tarinaa. Siinä jokaiselle säkeelle on oma viittomamerkki (alleviivattuina ohessa), jotka laulun edetessä viitotaan ja lauletaan. Seuraavilla kierroksilla sana kerrallaan jätetään laulamatta, mutta viittoma toistetaan. Kas metsämökin ikkuna. Sielt' tonttu ulos kurkistaa. Jänö laukkaa laputtaa ja oveen kolkuttaa. Auta, auta, pyydän sua. Metsämies kun vaanii mua. Sulle suojan tarjoan, siis kätes ojenna!” (Kangas & Vartiainen, 2019.)

Tavoitteena: laulun ja ohjelmoinnin yhdistäminen, ilo ja yhteenkuuluvuus

Tarvittavat välineet: puupalikoita linnan rakentamiseen, Alias-sanaselityskortit, piirustuspaperia, värikyniä

Lähteet:

Vartiainen, J., & Kangas, J. 2019 Ohjelmoinnin ABC varhaiskasvatuksessa

Geometriaa ja robottijumppaa, varhaiskasvatus

17.10.2022

Uudet varhaiskasvatussuunnitelman perusteet (2018) määrittävät, että tieto- ja viestintätekniset taidot sekä monilukutaidon osa-alueet ovat tulevaisuuden osaamista, ja näihin teemoihin tutustuminen lasten kanssa tutkien ja leikkien edistää lasten koulutuksellista tasa- arvoa.

Varhaiskasvatuksessa tavoitteena ei ole opettaa lapsia koodaamaan valmiita ohjelmia, vaan tukea heitä ihmettelyssä, tutkimisessa ja matemaattis-loogisen ajattelun kehityksessä, niin että he perusopetukseen siirtyessään voivat tasavertaisesti edelleen kehittää osaamistaan ohjelmoinnissa.

Ohjelmointi ja sille oleellinen algoritminen ajattelu ovat taitoja oppia uusia ajattelun tapoja. Ohjelmointi ja koodien luominen, eivät kuitenkaan ole tässä itsetarkoitus, vaan väline, jonka käyttöön voidaan pedagogisesti sisällyttää muita sisältöalueita ja ajattelun alueita, kuten matemaattis-looginen ajattelu, luova ajattelu, ongelmanratkaisutaidot sekä leikillinen ajattelu ja jaettu merkityksenanto.

Tuntisuunnitelma luo pohjaa ohjelmoinnin alkeisiin, tutustuttaen lapset ohjelmontikieleen symbolien avulla. Tunnin aiheena on geometriset muodot, joiden avulla tehdään Robotti-jumppa ja omat jumppamuotorobotit. Tavoitteena on tutustua geometrisiin muotoihin nimeämisen kautta, leikkien ja yhdessä luoden liikunnallinen tuokio sekä hyödyntäen kädentaitoja ja lasten omaa mielikuvitusta ja luovuutta. Tunneilla harjoitetaan myös muistia ja harjoitellaan yhteistyötaitoja.

Tunnin kulku: Kokoonnutaan piiriin (paikkana jumppasali). Piirin keskellä on muotolaatat (yksi kutakin). Opettajalla on pussi, jonka sisällä on pieniä muotolaattoja esim. jostain muotopalapelistä tai pelistä. Itse käytän tässä Pirate shapes pelin laattoja. Käydään ensin kaikki muotojen nimet läpi, jonka jälkeen lapset saavat vuorollaan ottaa sokkona muodon pussista ja nimetä sen sekä viedä oikean muotolaatan päälle piirin keskelle. Tämän jälkeen jaetaan lapset ryhmiin ja jokainen ryhmä saa kerätä itselleen yhden kutakin muotolaatoista. Muotolaattoja apuna käyttäen piirretään jokainen muoto paperille. Tämän jälkeen ryhmät asettuvat jonoon salin toiseen päähän, piirretty paperi asetetaan jonon viereen. Muotolaatat viedään toiseen päähän salia. Tarkoituksena on viestijuoksu, niin että jokainen lapsi juoksee vuorollaan muotolaattojen luokse, ottaa yhden muotolaatan ja tuo sen paperille, johon on piirretty muodot. Voittaja joukkue on se, joka on ensimmäisenä saanut kaikki muodot paperille. Tämän jälkeen kerrotaan lapsille, että he saavat olla

robotteja, jotka liikkuvat vain yhdessä sovituilla tavoilla. Kysytään lapsilta muotolaattoja apuna käyttäen, miten liikutaan minkäkin muotolaatan kohdalla. Harjoitellaan liikkeitä ja niiden yhdistämistä muotoihin. Kun lapset pääsevät jyvälle, niin opettaja tekee pidemmän sarjan muodoista, jonka jälkeen käydään sarja läpi liikkeitä tehden.

Tunti 2: Seuraavana aamuna koitetaan muistetaanko edellisenä päivänä luotu robottijumppa ja jumpataan sen tahdissa päivä käyntiin. Tämän jälkeen lapset saavat piirtää muotolaattoja apuna käyttäen muodot paperille, leikata muodot ja taiteilla niistä omat jumppamuotorobotit. Robotteja tehdessä kertaillaan muotojen nimiä. Lapsille voi tehdä lisätehtävän, jossa robotit väritetään ohjeen mukaan, esimerkiksi väritä kaikki kolmiot keltaiseksi, neliöt siniseksi ja ympyrät vihreäksi jne. Lopuksi lapset saavat piirtää robotin alle oman ”muotojumpan” eli käytetään edellisellä tunnilla opittua muotojumppaa. Nyt lapset voivat itse miettiä mikä liike vastaa mitäkin muotoa. Jumppamuotorobotteja voidaan käyttää myöhemmin aamujumpissa tai esimerkiksi odottelutilanteissa, jolloin lapsi voi opettaa oman jumppansa kaverille tms.

Vinkkejä ja ideoita suunnitteluun hain sivustolta: Ohjelmoinnin ABC varhaiskasvatukseen Jonna Kangas ja Jenni Vartiainen 2019

Beebot ohjelmointi ja matematiikan toiminnallisia tehtäviä, esiopetus

Satu Porkka, CC BY-SA 4.0

6.1.2020

Beebot ohjelmointi ja matematiikan toiminnallisia tehtäviä
Tuntisuunnitelma esiopetukseen

Beebot ohjelmointi toimii niin itsenäisenä harjoituksena, kuin myös voi olla myös osana eri oppimiskokonaisuuksia. Tässä suunnitelmassa Beebot johdattelee lapsen yhdessä parinsa kanssa matematiikan toiminnallisiin harjoituksiin, joissa on mukana ohjeen mukaan toimimista. Harjoituksissakin on siis ohjelmointia ja algoritmista ajattelua.

Asetetaan Beebotin matolle neljää eri kuva-aihetta. Lapset saavat vetää kuvakortin, ja sitä vastaavaan kuvaan matolla täytyy Beebotin päästä. Koodataan lasten valitsema reitti vaihe vaiheelta nuolinäppäimillä ja kokeillaan. Kun Beebot saadaan kulkemaan kohteeseen, lapset saavat kuvatunnukselle merkityn tehtävän suorittaakseen. Beebotin koodaus motivoi lapset harjoituksiin. Parit pääsevät tehtäviä suoritettuaan aina vuorollaan koodaamaan Beebotin uuteen kohteeseen, ja edelleen saavat uuden tehtävän. Opettaja sijoittuu ohjaamaan koodausta, mutta pääsee tarkastelemaan tehtäväpisteellä suoriutumista, kun on ottamassa parin uuteen koodaustehtävään. Yhdessä lasten kanssa samalla arvioidaan tehtävässä onnistuminen.


Lähde: Lukiloki -seminaari, Oulu.

Opetellaan muotoja, rakentelua, lukumääriä ja sarjoja pistetyöskentelynä parittain. Tarjotaan toiminnalliseen työskentelyyn sopivaa välineistöä, kuten tangram-palat, muotolaatat, muoviset pikkueläimet sarjoittamiseen ja multilinkit.

Tavoite ohjata lapset yhteistyöhön parin kanssa, rakentelemaan mallista, muodostamaan sarjoja ja keksimään omia kuviomuodostelmia muotolaatoilla. Pyritään harjoittamaan ongelmanratkaisua ja suunnitelmallisuutta toiminnallisilla harjoituksilla. Tavoitteena oppijan osallisuuden ja tekijyyden vahvistaminen.

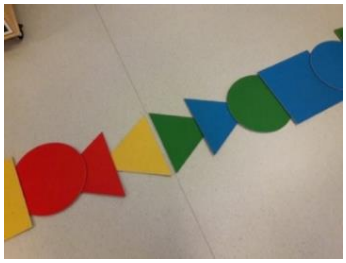
- tiedekasvatus
  Liikkuminen ympäristössä, ympäristön järjestäminen, ongelmanratkaisua, yhteistyö parin kanssa

- teknologiakasvatus
  Bee-botin käyttäminen, koodaus, rakentelu, symmetria, muodot, suunnat

- matematiikka
  Lukumäärän laskeminen, geometriset muotojen tunnistaminen, symmetria, sarjoittaminen, matemaattiset käsitteet

Tehtävä 1.
Peilikuva, symmetria

Pari asettuu vastakkain lattialle välissään viiva. Toinen parista alkaa muotolaatoilla (neliö, kolmio, ympyrä ja suorakulmio) rakentaa muodostelmaa yksi muoto kerrallaan, ja toinen seuraa pistämällä vastaavan muodon peilikuvaksi viivan toiselle puolelle. Kun peilikuva saadaan valmiiksi, vaihtuu osat ja vuorostaan toinen parista alkaa rakentamaan kuviota ja parinsa seuraa. Ohjataan käyttämään yksi muotolaatta kerrallaan ja tarkistamaan parilta ovatko kuviot oikeassa kohdassa.

2. tehtävä
Sarjoittaminen

Kartonkipohjilla kolme valmista sarjaa, jotka pari ensin ratkoo yhdessä, miten sarjaa jatketaan. Seuraavaksi lapset laativat toisilleen vuorotellen sarjatehtäviä. Sääntö käyttää tietty määrä pikku esineitä, esimerkiksi neljä esinettä.

Sarjoittaminen siihen tarkoitetulla välineistöllä, eli erivärisillä muovisilla pikkueläimillä, saattaa lapsilla rönsyillä, joten sääntö käyttää vain tiettyä määrää on paikallaan, jotta tehtävä hahmotetaan. Lapset tarvitsevat mallinnusta tehdä sarjoja. Jotta lapset kykenevät omatoimiseen työskentelyyn, sarjoitusta on harjoiteltava aikaisemmin ennen tätä pistetyöskentelyä.

3. tehtävä
Tangram-paloilla rakentaminen

Pari rakentaa rinnakkain kuvamalleista tangram -harjoituksia. Tehtävä kehittää muotojen ja koon hahmottamista sekä suuntatajua.

4. tehtävä
Rakentelu kuvamallista

Multilink -palikoilla rakentelua. Lapset ottavat rakentelumallin kolmiulotteisesta piirroskuvasta. Kuvaan merkitään myös lukumäärä, joka kertoo kuinka monta palikkaa rakentelussa on yhteensä. (Esiopetuksen laskutaito- kirja). Tehtävässä harjoitellaan suuntia, värien yhdistämistä ja lukumäärän hahmottamista. Tehtävä tehdään joko rinnakkain parin kanssa, tai pari voi tehdä yhteistyötä rakentelussa.

Tehtäväkokonaisuus toimii pienen ryhmän 3 – 4 parin harjoituksena, jolloin yhdellä pisteellä työskentelyyn käytetään sopiva aika, ja vältetään odottelua. Lapsilla voi olla aiempaa kokemusta samankaltaisesta koodauksesta esimerkiksi ScratchJr ja Bee-Bot App peleistä. Heidän ohjelmoinnin osaamista voi tällöin hyödyntää, ja suunnitella yhdessä matolle reitin vaikeusastetta esim. esteillä.

Pistetyöskentely pareittain mahdollistaa lapsille myös vertaisryhmältä oppimista ja palautteen saamista. Myöhemminkin tilanteissa, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelua, voidaan saada aikaan yhteistyötä ja parityöskentelyä, sekä ohjeen ottamista vertaiselta, ja näin vahvistaa ohjeiden seuraamisen taitoa.

Ohjelmointia esikoululaisille, esiopetus

Heli-Similä Saukkonen, CC BY-SA 4.0

11.6.2020

Ohjelmointia ESIKOULULAISILLE:

Opetusryhmä: Esikoululuokka, 6 oppilasta.

- Projektimme ohjelmointiin liittyen huipentuu nyt Beebotteihin tutustuen ja tehtävien tekeminen niiden avulla.

Suunnitelmalle varattu aika ja paikka:

- 3x 45min
- oma luokka, Kirkonkylän koululla

Opetettava aine:

- Algoritmisen ajattelun tukeminen ja suomenkieli. Tähän yhdistetään myös luonnontiede ja eläimet Suomen talvessa. Sosiaaliset taidot.

Tavoiteet:

- ohjelmointiin tutustuminen
- kuuntelemisen ja ohjeidenantamisen ja noudattamisen taidot
- yhdessä tekemisen taidot
- oman vuoron odottamisen taidot
- ajattelun ja ongelmanratkaisun taidot
- paikkakäsitteet
- vakiinnuttaa mieleen opittuja kirjaimia, äänteitä ja jo lukutaitoisille lapsille ylöspäin eriyttävästi lukemisen ja kirjoittamisen harjoituksia.
- tunnistaa suomen eläimiä ja niiden elämää ja jälkiä lumessa

Arviointi:

- Itse- ja vertaisarviointi.

- Välitön palaute onnistumisista opettajalta. Arviointi tapahtuu koko projektin osalta kokonaisuutena ja se dokumentoidaan lapsen kasvun kansioon valokuvin ja lapsen kommentein.
- toimintojen jälkeen yhteinen keskustelu, ja mielipiteet harjoituksista

Työskentelyvälineet ja opetusmenetelmät

- Rugged Robot auto
- 6klp beebot robotteja ja niihin kuuluvat alustat
- leikkieläimiä
- tehtävämonisteet ja kynät

Oppitunti 1:

Johdatus aiheeseen:

- olemme tutustuneet ohjelmointiin leikkien jo parin viikon ajan. Lapset ovat ohjelmoineet opettajaa ja toisiaan , joten perusajatus on jo selvillä. Olemme tehneet ohjelmointiratoja käyttäen erilaisia nuolimerkkejä, ja pelanneet erilaisia pelejä samoilla nuolimerkeillä(esim.ihmis kimble)

Tunnin aloitus:

Lapset saavat tehtävämonisteen,jossa ruudukkoa pitkin kuljetaan ohjeiden mukaan ja päädytään tiettyyn pisteeseen[esiopetusmateriaali].

Tämän jälkeen opettaja nostaa esille Rugged Robot auton, joka herättää lasten kiinnostuksen.

Yhdessä pohditaan mitä tapahtuu jos painetaan nappuloista ja leikkiluetaan nuolien suuntia.

Kaikki saavat kokeilla autoa ensin painelemalla nappuloita ja sen jälkeen merkitään paikka minne lapset yrittävät auton ohjata.

Tunnin arviointi:

- Keskustelua tunnin kulusta ja ohjelmoinnista. Onnistuinko, mikä oli mukavinta, hankalinta/ ikävintä? Yhteinen arvosana tunnin onnistumiselle.

Oppitunti 2: **Beebotit**

Beebotit otetaan esille ja lapset saavat valita omalle botilleen haluamansa “puvun”, jotta tunnistaa omansa.

Aluksi vapaata tutustumista botteihin ja niiden toimintaan, tunnin lopuksi pareittain työskentelyä bottien alustoilla. Toinen kertoo minne botin pitää mennä ja toinen ohjelmoi.

Tunnin arviointi:

- Keskustelua tunnin kulusta ja ohjelmoinnista. Onnistuinko, mikä oli mukavinta, mikä ikävintä/hankalinta? Yhteinen “arvosana” tunnin onnistumiselle.

Oppitunti 3: BEE BOTIT ja itsevalmistetut tehtävä matot

Haetaan Bee botit jalaitetaan niille “puvut” päälle, jotta oma botti tunnistetaan.

Opettaja on teipannut lattiaan valkoiselle paperille tulostettuja, tyhjiä tehtävämattoja. Matolle on sijoitettu leikkieläimiä, samat eläimet löytyvät myös merkityiltä pöydiltä piilotettuina jugurttipurkin alle. Eläimen alla on koodilappu(väri), joka ohjeistaa lapsen oikealle pöydälle. Lapsille jaetaan tehtävämonisteet, joissa on eläinten jälkiä ja tyhjä viiva.

Lasten tulee ohjelmoida Bee bottinsa eläinten luokse, tunnistaa eläimen jäljet paperista ja käydä kirjoittamassa eläimen nimi viivalle. Pöydällä on valmiiksi kirjoitettu lappu oikean eläimen luona, mikäli lapsi ei osaa vielä itse lukea/ kirjoittaa.

Jokaisen eläimen jugurttipurkista pöydältä löytyy myös kirjain ja numero.Numero kertoo kohdan tehtäväpaperissa, johon kirjain kuuluu, ja näin muodostuu salasana K,O,O,D,I.

Tunnin ja koko projektin arviointi:

- Keskustelua oppimiskokonaisuudesta ja sen sisällöistä. Onnistuinko jossain, mikä oli mukavinta, mikä ikävintä/hankalinta? Mitä opin? Mitä haluaisin vielä oppia?Yhteinen “arvosana” oppimiskokonaisuuden onnistumiselle.

Oppimiskokonaisuuden koonti:

Poikavaltaisessa ryhmässäni autot ja tekniikka kiinnostavat kovasti ja oletan, että Rugged Robot kiinnostaa kovasti. Yhteistyöhön ja oman vuoron odottamiseen tarvitaan varmasti kannustusta, koska mukana on erityislapsia, mutta olemme harjoitelleet sitä kovasti, joten luulen senkin onnistuvan pikku porukassa. Kannustamme toisiamme ja mietimme yhdessä kuinka haluttuun pisteeseen autolla päästään Kaikessa toiminnassa hyödynnetään jo opittuja asioita ja tällä luodaan pohjaa tulevaa oppimista varten. Boteilla lapset pääsevät työskentelemään itsenäisemmin, mikä motivoi omalta osaltaan. Salasanojen ratkaiseminen luokassani on mieluista puuhaa ja sen ratkaiseminen on haaste, johon he tarttuvat varmasti

Lähteet:

https://www.innokas.fi/wp-content/uploads/2018/03/Terhin_luento130320

18.pdfhttps://blog.hamk.fi/koodausta-kouluarkeen/eskari-ykkosten-bee-bot-koodaus

http://hdl.handle.net/10138/301730

Ohjelmointia esioppilaiden kanssa, esiopetus

11.6.2020

Ohjelmointia esioppilaiden kanssa

Ryhmässäni on 13 eskaria. Jaan ryhmän puoliksi, jolloin toinen puoli ryhmästä tekee päivähoitajan kanssa muuta yhdessä sovittua ja toinen puoli ryhmästä osallistuu tähän harjoitukseen. Seuraavana päivänä vaihdamme ryhmät.

Olemme jonkin verran käyttäneet ohjelmointia aikaisemminkin ja tutustuneet mm. Bee-botien toimintaan. Nyt tarkoituksenamme on ohjelmoida Bee-Botit kulkemaan alustalla etsien kirjaimia ja sanoja. Yhdistän tässä ohjelmoinnin, koodauksen ja äidinkielen.

Oppimistavoitteet:

- Yhdessä tekemisen tukeminen ja kasvaminen
- sijaintikäsitteiden harjoittelu
- koodauksen opettelu
- kielen kehitys: kirjainten ja äänteiden kertaus, sanahahmojen tunnistaminen
- Lukutaitoisille ylöspäin eriyttäen lauserakenteeseen tutustuminen, sanahahmojen tunnistaminen ja löytäminen, lukemisen ja kirjoittamisen harjoittelu

Tarvittavat välineet:

- Bee-Botit
- Muovitaskullinen Bee-Bot-alusta ja itse kartongista tehtyjä Bee-Bot-alustoja
- Värillisiä muovikartioita
- Kirjainkortteja, kuvakortteja ja sanakortteja
- vihot ja lyijykyniä

Toiminnan kulku:

Leikimme aluksi ohjelmoinnillisen leikin. Laitan erivärisiä kartioita riviin. Mietimme kullekin värille jonkin liikkeen. Painelen eri järjestyksessä kartioita, jolloin lapset tekevät nopeasti väriin kuuluvan liikkeen. Lapset toistavat liikkeen niin monta kertaa, kuin painan kartiota. Tämän jälkeen lapset saavat tulla painelemaan kartioita.

Otan Bee-Botit esille. Tämä jo kokemukseni mukaan innostaa lapsia. Minulla on lattialla kolme Bee-Bot-alustaa. Olen tehnyt alustat huomioiden eriyttämisen. Yhdessä alustassa on kirjaimia, toisessa on kuvia ja kolmannessa on sanoja. Jaan lapset taitotason mukaan sopiville matoille. Kirjainmatolla sanon lapsille kirjaimen, joka heidän pitää etsiä matolta. Kun he ovat löytäneet kirjaimen, menevät he kirjoittamaan kyseisen kirjaimen omaan vihkoonsa. Lapset voivat myös antaa toisilleen ohjeita, minkä kirjaimen he etsivät. Mikäli tämä rupeaa sujumaan, laitan viereen kuvakortteja nurinpäin. Lapset saavat vuoron perään nostaa kuvakortin ja miettiä, millä kirjaimella kyseinen asia alkaa ja ohjelmoida Bee-Bot oikean kirjaimen kohtaan. Kannustan lapsia keskusteluun ja yhteistyöhon.

Toisella matolla on kuvakortteja. Bee-Bot-alustan edessä on kirjainkorttipakka nurin päin. Lapset saavat vuoron perään nostaa kirjaimen ja etsiä kyseisellä kirjaimella alkavan kuvan alustalta ja ohjelmoida Bee-Bot oikeaan ruutuun. Tämän jälkeen lapsi saa mennä kirjoittamaan kirjaimen vihkoonsa. Mahdollisesti ajan sallimissa puitteissa hän voi myös piirtää kyseisen kuvan kirjaimen viereen. Mikäli on aikaa, saavat myös nämä ryhmät vaihtaa keskenään alustoja.

Kolmannella alustalla on lukeville lapsille sanakortteja. Katsomme monisteesta ensin läpi lauserakenteen. Minulla on moniste, jossa on eri värillä eri sanaluokat. Ensin on kuvan kanssa kuka. Tämän jälkeen punaisella on mitä tekee ja sitten sinisellä missä. Bee-Bot-alustalla on lokeroissa kortteja, joissa jokaisessa on nämä värit. Lapsi saa itse valita, miten Bee-Botin ohjelmoi, mutta hänen on edettävä lauserakenteen mukaisesti etsien ensin tekijän, joka on kuvana alustalla. Tämän jälkeen hänen on etsittävä mitä tekee. Tämä siis lukee punaisella ja lopuksi vielä sinisellä kirjoitettu missä. (Huom! Myöhemmin, kun tätä on harjoiteltu, lisään alustalle vielä millainen ja milloin.) Kun lapsi on ohjelmoinut virkkeen Bee-Botilla, menee hän kirjoittamaan virkkeen vihkoonsa.

Lopuksi mietimme lasten kanssa, miltä toiminta vaikutti, mistä he pitivät ja mistä eivät, mitä he oppivat ja mitä haluaisivat jatkossa lisää.

Lähde: Lukilokikoulutus. Valinnaiset opinnot. Ohjelmointi osana tuottavaa kirjoittamista ja luetun ymmärtämistä. Jyväskylän yliopisto ja Niilo Mäki instituutti. Lähiopetuspäivä Jyväskylässä 7.9.2019

Koodauksen alkeet, esiopetus

Tiina Lähteenaro, CC BY-SA 4.0

26.3.2021

Tiina Lähteenaro

Koodauksen alkeet ja matematiikkaa toiminnallisesti

Tuntisuunnitelma esiopetukseen

Esiopetuksessa voi hyvin harjoitella koodauksen alkeita. Tämän tuntisuunnitelman (noin 2h) tavoitteena on oppia koodauksen perusperiaatteita, yksinkertaisten käskyjen antamista sekä toiminnallisuutta niin itse toimien kuin esimerkiksi BlueBotia ja/tai BeeBotia käyttämällä.

Tavoitteena on harjoitella toimimista myös parin kanssa. Ohjelmointi ja koodaus toimivat tässä suunnitelmassa niin itsenäisinä harjoituksina, opettajajohtoisena työskentelynä, toiminnallisena oppimisena sekä niitä on mahdollista hyödyntää osana eri oppimiskokonaisuuksia.

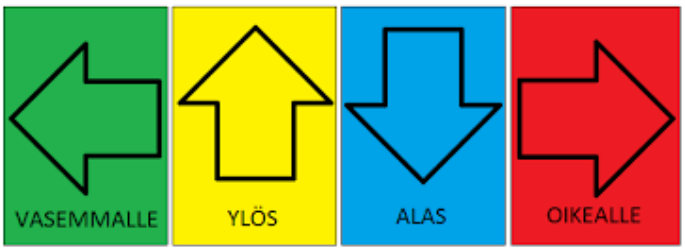
Opettaja voi arvioida lasten oppimista erilaisilla toiminnallisilla tehtäväpisteillä. Lapset pääsevät myös harjoittelemaan oman oppimisen arviointia sekä mahdollisesti antamaan palautetta myös kaverille, esimerkiksi koodausohjeiden antamisesta. Esiopetuksessa harjoitellaan arviointia hyvin kevyesti, esimerkiksi tehtävässä onnistumista tai voidaan miettiä, mitä jäi mieleen eli mitä tuli opittua, tai miten kaveri antoi ohjeita (oliko ohjeita helppo noudattaa, tuliko ohjeita kerralla sopivasti jne.) Tunneilla harjoitellaan niin ohjelmoinnin ja koodauksen alkeita kuin algoritmista ajattelua, geometriaa (muodot), kirjaimia, numero-lukumäärävastaavuutta jne.)

Ensimmäinen tunti:

Tarvikkeet
1. piste: paperi, jolla esim. 3x3 tai 4x4 ruudukko, pikkuauto/leluhahmo tms. ruudukkoon sopiva tavara, koodauskortteja (nuoli eteen, taakse, vasemmalle, oikealle)
2. piste: Lattiaan teipattu 3x3 tai 4x4 ruudukko, noppa jossa jokaisella nopan sivulla koodauskortit (nuoli eteen (näitä 2 sivulla), taakse, vasemmalle, oikealle, stop.
3. piste: Ipadit joissa BeeBot-pelisovellus.

Lapset jaetaan 3 ryhmään (esimerkiksi sopiva määrä yhdelle pisteelle on max.4 lasta). Kukin ryhmä tekee vuorollaan yhdellä kolmesta pisteestä, toiminta-aika noin 15min.

1.piste: harjoitellaan ruudukon ja leikkiauton ym. hahmon avulla koodausohjeiden antamista ja liikkumista ruudukossa. Lähtöruudukko esim. alhaalla vasemmalla. Nostetaan pinosta ennalta sovittu määrä kortteja (esim. 4kpl). Liikutetaan autoa korttien määrittämien koodausohjeiden mukaan; esimerkiksi: eteen, käännös oikealle, eteen, taakse. Mihin auto pysähtyy?
Esimerkki korteista:



Ensin opettaja näyttää periaatteen, tämän jälkeen lapset pääsevät harjoittelemaan itse. Työskennellään esim. pareittain; toinen nostaa kortit, toinen liikuttaa autoa. Pidetään korttien määrä mahdollisimman pienenä, taitavimmat voivat alkaa nostaa esim. 6-8-10 korttia pakasta.

2. piste: Lattiaan teipattu ruudukko ja noppa, jonka sivuilla samanlaiset ohjekortit kuin 1. pisteellä. Pari heittää noppaa esim. 4 kertaa (6-8-10), toinen hyppii ruudukossa korttien mukaan. Voidaan tehdä esim. heitto ja liikkuminen, seuraava heitto ja liikkuminen tai ottaa toinen pakka kortteja, järjestää nopan heittojen mukaan järjestykseen ja sen jälkeen liikkua kaikki kerralla.

3. piste: IPadin BeeBot pelisovelluksella harjoitellaan koodausohjeiden antoa. Pelikentät vaikeutuvat pelin edetessä.

2. oppitunti

Tarvikkeet:

BeeBot-koodausmattoja (esim. lukumäärämatto, muotomatto, aakkosmatto, numeromatto jne. erilaisia valmiita mattopohjia tai itse tehtyjä), BeeBot ja/tai BlueBot lattiarobotteja 1kpl/matto

IPadit BeeBot-sovellus

Vihkot ja kynät

Aakkoskortteja, numero- ja lukumääräkortteja, muotokortteja, koodauskortit (nuolikortit)

Jaetaan ryhmä jälleen pieniin ryhmiin, yksi ryhmä per matto. Toiminta-aika noin 15min per piste.

Koodausmattoja ja BeeBot- tai BluBot- mattoja mieluiten ainakin 4, joissa em. asiat, lisäksi voi käyttää valmiita mattopohjia myös esim. maatila- tai kaupunkimatto.

Tehtävät:

1. Aakkosmatolla lapsi nostaa aakkoskortin ja koodaa robottinsa liikkumaan ko. ruutuun.

2. Numeromatolla lapsi nostaa numerokortin (numeromatolla ainakin numerot 0-10 tai 0-20) ja koodaa robottinsa liikkumaan ko. ruutuun. Tämän jälkeen pari etsii omalta lukumäärämatoltaan kyseisen ruudun ja ohjeistaa oman robottinsa liikkumaan siihen. Tämän jälkeen pari vaihtaa ”paikkaa”.

3. Muotomatolla lapsi nostaa muotokortin ja koodaa robottinsa liikkumaan ko. ruutuun. Tämä toistetaan esim. viisi (5) kertaa. Toinen parista kerää muotokortit riviin, jonka jälkeen muodot piirretään vihkoon samassa järjestyksessä (esim. kolmio-kolmio-neliö-ympyrä-kolmio). Samalla parit nimeävät muodot toisilleen. Tämän jälkeen vaihdetaan taas tehtävässä toisinpäin.

4. Mikäli käytössä valmiita mattopohjia, työskennellään niissä vuorotellen pareittain: toinen kertoo mihin ruutuun robotti liikkuu, toinen ohjelmoi.

5. Varalla Ipadit ja BeeBot -sovellus, jota voi pelata.

Oppitunneilla harjoitetaan:

Koodaus ja ohjelmointi

Algoritminen ajattelu

Ongelmanratkaisu

Muodot

Numerot

Lukumäärät

Aakkoset

Suunnat

BeeBotin ja BlueBotin käyttäminen

Sarjoittaminen

Tämä tuntisuunnitelma on täysin itse keksimäni ja olen sitä toteuttanut useana vuonna esiopetuksessani. Suunnitelma sopii myös hyvin alkuopetuksessa käytettäväksi.

Paikkakäsitteiden ja ongelmanratkaisutaitojen harjoittelua BeeBot-roboteilla, esiopetus

28.3.2021

TUNTISUUNNITELMA (OSA II: Oppijan näkökulma)
Ohjelmointia esikoululaisille

ESITIEDOT

Ohjelmointi harjoitukset ovat suunniteltu esikouluikäisille. Tuntien aikana tutustutaan ohjelmointiin erilaisten tutustumisharjoitusten kautta.

OPPIMISTAVOITTEET

Tuntien tavoitteena tutustua ohjelmointiin ja sen perusajatteluun. Harjoittelussa käytetään myös Bee-bot -lattiarobotteja. Tavoitteena harjoitella paikkakäsitteitä sekä ongelmanratkaisutaitoja. Bee-bot harjoituksen tavoitteeena harjoitella kirjain- ja lukuharjoittelua.

TYÖVÄLINEET

Harjoituksia varten tarvitaan lattiakuviot, paperi, kynä, pelialusta (ruudukko jossa kirjaimia) sekä Bee-bot -robotit. Ennakkovalmistelu: opetajan tutustuminen Bee-bot -lattiarobotteihin ja niiden toimintaan sekä ruuudukon ja käskyjen valmistelu ja tulostus.

ARVIOINTI

Välitöntä sanallista palautetta lapsille. Tuntien lopuksi yhteenveto ja lasten oma arviointi tunnista, sen harjoitteista sekä työskentelyn sujuvuudesta.

OPPITUNTI 1:

Aloitetaan tunti johdattamalla aiheeseen. Mitä ohjelmointi on? Miten me voimme ohjelmoida? Mitä voidaan ohjelmoida? Ensimmäisen tunnin harjoitteet voidaan toteuttaa ilman tietokonetta.

Harjoitus 1: Kapteeni käskee -leikki. Toimitaan pareittain. Pari antaa ohjeita ja toinen tekee ohjeiden mukaan. Vaihdetaan rooleja. Harjoituksen avulla tutustutaan ohjeiden antamiseen, ja siihen mitä seurauksia ohjeiden toteutuksesta on.

Harjoitus 2: Opettaja robottina. Opettajaa ohjataan määrätyn aarteen luo, esimerkiksi hakemaan kirja hyllystä. Opettajalle annetaan lyhyitä ohjeita, esim. kävele kolme askelta eteenpäin, käänny oikealle, jne…. Huomataan, että kaikkia käskyjä ei voida

välttämättä toteuttaa, jos esimerkiksi kävelee seinää päin. Ohjeiden tulee olla myös selkeitä ja ymmärrettäviä (ei esimerkiksi kävele tuonne kolme askelta).

Harjoitus 3: Kaverin ohjaaminen lattia merkeillä. Harjoitus tehdään pareittain. Toinen on ohjaaja ja toinen kulkee radalla silmät kiinni. Ohjeita annetaan taputtamalla olkapäihin (oikea olkapää = käännös oikeaan, vasen olkapää = käännös vasempaan, molemmat olkapäät = askel eteenpäin).

Tunnin arviointi: Keskustellaan tunnin kulusta ja ohjelmoinnista. Mikä oli mukavinta, mikä hankalinta? Opettaja antaa sanallista palautetta koko ajan tunnin kuluessa.

OPPITUNTI 2:

Toisella oppitunnilla harjoitellaan ohjelmointia Bee-bot -robottien avulla. Aluksi on hyvä harjoitella robottien perustoiminnot ilman erillistä harjoitusta. Miten robotti saadaan liikkeelle? Millaisia käskyjä sille voidaan antaa? Harjoituksessa käytetään yksinkertaisia komentoja: liiku yksi ruutu eteenpäin, liiku yksi ruutu oikealle, liiku yksi ruutu alaspäin, liiku yksi ruutu vasemmalle.

Harjoitellaan ohjelmointia ja algoritmista ajattelua ruudukolla. Komennoilla muodostetaan vaiheittainen toimintaohje eli algoritmi. Algoritmi suoritetaan yksi komento kerrallaan ja Bee-bot liikkuu komennon mukaisesti.

Pari saa yhden 5x5 ruudukon, jossa on kirjaimia ruuduissa sekä käskyjä, joita suorittaa. Bee-botia liikutetaan ohjeiden mukaisesti. Ja se saapuu aina yhden kirjaimen luo. Pari kirjoittaa kirjaimen monisteelle. Kirjaimista muodostuu sanoja. Muodostettu sana luetaan ääneen.

Tunnin arviointi: Keskustellaan tunnin kulusta ja ohjelmoinnista. Mikä oli mukavinta, mikä hankalinta? Opettaja antaa sanallista palautetta koko ajan tunnin kuluessa.

Ohjelmointia toiminnallisesti robottileikkien avulla, esiopetus

Anna Långström, CC BY-SA 4.0

7.11.2021

Aihepiirin valinta

Suunnittelen opetustuokiot eskariryhmälleni, jolla ei ole aikaisempaa kokemusta ohjelmoinnista. Lähdemme tutustumaan ohjelmointiin toiminnallisesti, sillä se on ryhmäni lapsille ja minulle mieluinen tapa työskennellä. Koska lapsilla ei ole aikaisemmin ollut ohjelmointia, lähestymme sitä leikin kautta. Robotit ovat lapsista mielenkiintoisia ja he ovat rakennelleet niitä legoista ja palikoista, joten siitä saamme hyvän aasinsillan ohjelmoinnin alkeisiin.

Tutustumme robotteihin ja niiden toimintaan sekä niiden käyttämään kieleen ensin keskustellen, sitten leikkien ja lopuksi Beebottia käyttäen. On tärkeää, että jokainen kokee osallisuutta ja pääsee tekemään. Jaan kaksi tuntia kahdeksi eri opetustuokioksi, koska lapset jaksavat näin paremmin. Rohkaisen jokaista osallistumaan ja toimimaan yhdessä.

Oppimistavoitteisiin kuuluu ohjeiden kuuntelu, ongelmanratkaisu, empaattinen ajattelu, toisen asemaan asettuminen, yhteiseen toimintaan sitoutuminen, oman vuoron odottaminen, vuorovaikutustaidot sekä hyvä kielellinen ilmaisu.

Arviointi

Oppituokioiden arviointi tavoitteisiin nähden tulee suoraan lapsilta tuossa hetkessä. Ovatko he innolla mukana, lähtevätkö ilolla kokeilemaan ja osaavatko ottaa toisensa huomioon tehtäviä suorittaessaan. Lapset myös kertovat palautetta aikuiselle välittömästi, oliko joku kivaa vai ei. Voimme käydä yhteisesti vielä läpi oppituokioiden jälkeen, miten lapset arvioivat tehtäviä. Meillä on vakiintunut peukku ylös- peukku alas jokaviikkoiseksi palautteeksi eskariviikosta, joten samaa voimme käyttää tähänkin. Saan itselleni siitä arvion, kuinka suunnittelemani kokonaisuus on toiminut ja miten se suhtautuu omaan arviooni samoista kohdista.

Toteutussuunnitelma

Olen jakanut lapset pienryhmiin ja toinen puolikas on ulkona. Aloitamme keskustelemalla aamupiirissä roboteista. Miltä ne näyttävät ja miten ne toimivat, millaista kieltä ne käyttävät ja miten niitä ohjataan. Sen jälkeen leikimme robotteja. Annan lapsille ohjeita, kuinka he toimivat kuin robotit käskyjä noudattaen. Ota kaksi askelta eteenpäin, pysähdy. Hyppää ilmaan. Nosta kädet ylös, laske ne alas. Mene kyykkyyn. Nouse ylös. Kun tämä sujuu, saavat lapset ohjata pariaan samalla lailla. Ja sitten vaihdetaan niin että molemmat saavat olla robotteina.

Lopuksi esittelen vielä Nestori Robotin lapsille. Kerron, millaista Nestorin kieli on ja miten se toimii (neliö tarkoittaa askelta eli kävelemistä, ympyrä tasajalkahyppyä, kolmio osoittaa suuntaa ja tähti tarkoittaa vilkuta). Teen etukäteen kartongista valmiit merkit ja laminoin ne, joista saamme tehdä Nestorin komentoja ilmoitustaulun reunaan. Kokeilemme Nestorin komennoilla toimimista yhdessä ensin lyhyellä

merkkijonolla ja lopuksi vähän pitemmällä. Sen jälkeen lapset saavat tulla tekemään komentoja toisilleen.

Välineinä tarvitaan robotin kuvia havainnollistamiseksi, tarpeeksi tilaa liikkua, kartongista tehdyt neliö, ympyrä, kolmio ja tähti useina kappaleina.

Seuraavana päivänä aamupiirissä muistelemme Nestorin komentoja ja keksimme lisää. Rakennamme lasten kanssa esteradan muovitörpöistä ja kokeilemme kulkea robotteina reitin läpi niin että lapset saavat ohjata ensin opea ja sitten toisiaan. Otan esiin Beebotin ja kerron lapsille, että sekin on robotti, joka tottelee käskyjämme, kun painamme nappeja. Esittelen Botin liikkeitä havainnollisesti nappeja selostaen. Voimme ohjelmoida Botille reitin ensin yksittäisinä komentoina läpi sen oman maton ja sitten niin, että se kulkee koko maton läpi yhdellä kertaa. Lopuksi Botti saa mennä saman törppöreitin kuin lapsirobotit. Lapset saavat laittaa reitin varrelle aarteita, joita Botti etsii. Kun opetustuokio on lopussa, Botti valitsee pyörien oman akselinsa ympäri ja pysähtyen jonkun lapsen kohdalle ensimmäisen uloslähtijän ja sitten seuraavan, niin että kaikki lapset ovat pukemassa.

Välineinä tarvitaan Nestorin merkit uudelleen, muovitörppöjä, Beebot-robotti, sen matto, pieniä aarteita Botin reitille.

Uskon, että jatkamme ohjelmointia Beebotin kanssa ja ilmankin, lapset myös siirtävät leikkeihinsä usein yhdessä tehtyjä juttuja. Voisimme kokeilla uusia juttuja Beebotin kanssa, esimerkiksi pelata palloa ja maalata botin kanssa. Lisäksi lasten kanssa tehtyä robottileikkiä voimme jalostaa eteenpäin sekä siirtyä tekemään legolabyrinttiä tai paperilla olevaa sokkeloa, jossa nuolin ohjataan reittiä läpi sokkelon.

Tuokioiden suunnittelussa olen käyttänyt apuna Ohjelmoinnin ABC varhaiskasvatuksessa-kirjaa (Jonna Kangas-Jenni Vartiainen).

BeeBot-robotteihin tutustumista, esiopetus

4.10.2021

Bee-Bot -robotit esiopetuksessa

Kohderyhmä: Esiopetusikäiset lapset (noin. 20 lasta)

Aihepiiri:

Bee-Bot -robotteihin tutustumista ja niiden ohjelmointia. Lapset myös suunnittelevat ja tuottavat oman maton Bee-Bot -robottien harjoittelemiseen. Bee-Bot -robottien käyttöä yhdistetään kirjaimiin ja sanojen harjoitteluun sekä kuvataiteelliseen ilmaisuun. Esiopetuksen laaja-alaisen osaamisen tavoitteita projektissa ovat tutkin ja toimin ympäristössäni, kielen rikas maailma ja ilmaisun monet muodot (Opetushallitus, 2016). Tutkin ja toimin ympäristössäni tavoitteista harjoitellaan matemaattisia taitoja leikin ja yhdessä toimimisen kautta (Opetushallitus, 2016). Esimerkiksi toiminnassa tulee harjoiteltua laskemista ja mittaamista. Lasten kanssa tutustutaan kirjoitettuun kieleen osa-alueesta kielen rikas maailma (Opetushallitus, 2016). Lapset harjoittelevat kirjaimia, tavuja ja sanoja. Ilmaisun monet muodot alueesta harjoitellaan kuvataiteellista ajattelua ja ilmaisua (Opetushallitus, 2016). Lapset suunnittelevat ja tuottavat käyttäen apunaan kuvataiteellista ajattelua ja ilmaisua. Näiden lisäksi lapset harjoittelevat ryhmässä toimista ja yhteistyötaitoja.

Materiaalit:

Bee-Bot- robotteja, kirjain alusta, kyniä, tusseja, paperia, tabletteja, liimaa, saksia, viivoittimia

Arviointi:

Opettaja antaa lapsille jatkuvasti palautetta heidän toiminnastaan esimerkiksi ryhmässä toimimisesta ja robotin ohjelmoinnin kehittymisessä. Opettaja kannustaa lapsia tekemään haastavampiakin tehtäviä. Tuntien lopuksi keskustellaan esimerkiksi: opinko jotain uutta, mitä opin, mikä oli tuttua, oliko mielenkiintoista, mikä oli kivaa tms. Lapset näyttävät peukulla ylös sekä voivat kommentoida lisäksi.

1. tunti

Tavoitteet:

Lapset harjoittelevat robottien ohjelmointia ja saavat niissä onnistumisen kokemuksia. Lapset harjoittelevat käskyjen antamista Bee-Bot -roboteille. Lapset harjoittelevat kirjaimia, tavuja ja sanoja.

Sisältö:

Ensimmäisenä tutustutaan opettajan ohjauksella Bee-Bot -robottien toimintaan. Käydään läpi, minkälaisia komentoja sille voidaan antaa ja miten se toimii. Lasten annetaan harjoitella lyhyitä komentoja.

Kun Bee-Bot -robottien toiminta on tullut tutuksi, jaetaan ryhmä pienempiin 2–3 hengen ryhmiin. Alustana on kirjaimia. (ks. kuva1). Ensimmäisenä toiset lapset ohjaavat lasta kulkemaan tietyn kirjaimen luokse. Kun lapset kehittyvät heitä voidaan käskeä muodostamaan tavuja ja sen jälkeen myös sanoja. (esimerkiksi oma nimi ja tuttuja sanoja, joita he tietävät.) Lasten taidon mukaan voidaan myös tehdä niin, että lapsi tekee sanan robotilla ja muut arvaavat, mikä sana siitä tuli. (Sanaan kuuluvat kirjaimen kohdalla robotin pitää pysähtyä pidemmän aikaan, joten tiedetään, että kirjain kuuluu sanaan). Opettajan tehtävänä on kierrellä ryhmissä antamassa lisää ohjeita ja

haastavampia tehtäviä sekä vastata lasten kysymyksiin antaen lasten kuitenkin ensin pohtia yhdessä ryhmän kanssa.

Tunnilla kannattaa ottaa huomioon, kuinka monta Bee-Bot -robottia on tarjolla. Tunti voidaan rakentaa niin, että puolet lapsista tekee muita tehtäviä toisessa huoneessa. Esimerkiksi kirjaimiin liittyviä tehtäviä.

(kuva1). Kirjainalusta

2. tunti

Tähän kannattaa varata noin 2–3-tuntia.

Tavoitteet:

Lapset suunnittelevat ja tuottavat alustat. Lapset harjoittelevat ryhmässä toimimista.

Sisältö:

Lapset jaetaan n.5 hengen ryhmiin. Lapset suunnittelevat omat alustat, joissa kuljetaan Bee-Bot -roboteilla. Lapset saavat käyttää omaa luovuuttaan ja keksiä minkälaisen haluavat. Opettaja voi tarvittaessa antaa vinkkejä ideointiin. (esimerkiksi joku voi tehdään numeroihin liittyvän tai liikuntaan). Lapset käyttävät kirjainalustaa mallina siihen, että robotti saadaan kulkemaan oikeanlain alustalla. (ruutujen tulee olla oikean kokoisia). Lapsille annetaan monipuolisia mahdollisuuksia toteuttaa alusta. (esim. piirtäminen, kuvien tulostaminen, kuvien ottaminen ympäristöstä)

3. tunti

Tavoitteet:

Lapset harjoittelevat pidempien käskyjen antamista robotille. Lapset harjoittelevat oman alustan esittämistä yhdessä muiden ryhmäläisten kanssa.

Sisältö:

Ensimmäisenä esitellään muille ryhmille omat tuotokset. Sen jälkeen omalla alustalla harjoitellaan Bee-Botin käyttöä 2–3 hengen ryhmissä. Myös vaihdetaan muiden ryhmien kanssa alustoja. Opettaja ohjeistaa pidempien käskyjen antamiseen robotille.

Robottien määrä voidaan huomioida niin, että jokainen viiden hengen ryhmä jaetaan 2–3 hengen ryhmiin. Puolet osallistuvat ensin ja toiset puolet tekevät jotain muuta.

Lähteet:

Opetushallitus (2016). Esiopetuksen opetussuunnitelman perusteet 2014. Tampere: Juvenes Print. Haettu osoitteesta: https://www.oph.fi/sites/default/files/documents/esiopetuksen_opetussuunnitelman_perusteet_2014.pdf

Kuva1: https://peda.net/hankkeet/digi/koulutus/ohjelmointi/materiaalit/m2/bee-bot

Ohjelmointiin tutustumista toiminnallisesti , esiopetus

30.9.2021

Aihepiirin valinta ja rajaus

Teen suunnitelman esiopetukseen ja toteutan opetuskokeilun koulumme esikoulussa. Espoon kaupungin esiopetuksen opetussuunnitelmassa teknologiakasvatus ja matematiikka liitetään opintokokonaisuuteen nimeltään Tutkin ja toimin ympäristössäni. Ohjelmoinnin alkeet palvelevat niin matemaattisen ajattelun ja esimatemaattisten taitojen kehittymistä sekä teknologiakasvatuksen tavoitteita.

Opetussuunnitelman mukaan esiopetuksessa tutustutaan tutkivaan oppimiseen havainnoimalla ja tutkimalla ympäristöä sekä kokeilemalla ja päättelemällä. Esiopetuksessa tavoitteena on vahvistaa pohjaa lasten matemaattisen ajattelun kehittymiselle ja matematiikan oppimiselle. Lapsia ohjataan kiinnittämään huomiota arjessa ja ympäristössä ilmenevään matematiikkaan ja teknologiaan.

Esiopetuksessa tutustutaan tutkivaan työtapaan ja lapsia rohkaistaan tekemään kysymyksiä ja etsimään niihin yhdessä selityksiä. Lapset opettelevat esimatemaattisia, ajattelun taitoja, kuten vertailua, luokittelua, sarjoittamista sekä havaintojen tai mittausten pohjalta saatujen tietojen järjestämistä. Esikoululaisille ei tarjota valmiita ratkaisuja, vaan heitä rohkaistaan tekemään päätelmiä ja keksimään ratkaisuja arjen ongelmiin sekä kokeilemaan ratkaisuja konkreettisesti.

Lapsia ohjataan myös havainnoimaan ympäristön teknologiaa. Lasten kanssa havainnoidaan ja pohditaan arjessa esiintyviä teknologisia ratkaisuja ja niiden ominaisuuksia ja toimivuutta. Lapset tutustuvat teknologiaan keksimällä, askartelemalla ja rakentamalla itse erilaisia rakenteita ja ratkaisuja eri materiaaleja hyödyntäen. Lapsia kannustetaan kuvailemaan tekemiään ratkaisuja.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppikokonaisuuden tavoitteena on tutustuttaa esikoululaiset ohjelmoinnin peruslähtökohtaan, ihmisen ja koneen vuorovaikutukseen vaiheittaisten toimintaohjeiden avulla. Lisäksi oppimiskokonaisuuden tavoite on pysähtyä miettimään, miten koneen ajattelu eroaa ihmisen ajattelusta. Keskeistä on sitoa käsite ohjelmoinnista ja algoritmisesta ajattelusta esikoululaisten kokemusmaailmaan. Kaikki opetus ja toiminta kokonaisuuden aikana on toiminnallista ja konkreettista.

Esikoulussa arviointi on formatiivista ja oppimisprosessia seuraavaa. Arvioinnilla on kahtaalle ulottuva rooli - toisaalta arvioinnin avulla suunnitellaan ja kehitetään toimintaa, toisaalta tuetaan lapsen kasvua, hyvinvointia ja oppimista. Tärkeää on havainnoida akateemisten tavoitteiden lisäksi lapsen vuorovaikutustaitoja, ryhmätyötaitoja sekä kielellisiä taitoja. Oppilas on itse aktiivinen osa oppituntia ja sen arviointia. Oppimiskokonaisuuden arviointi perustuu siis aikuisen omaan reflektioon, toiminnan seuraamiseen, oppimiseen ja lapsen omaan kokemukseen tuokiosta.

Tunnin kulku, rakenne, työskentelyvälineet ja opetusmenetelmät

Tunnin alussa lapsille kerrotaan, että tänään opetellaan juttelemaan tietokoneiden kanssa ja selvitetään, miten koneet ajattelevat. Tuokion aikana myös ollaan itse vuorotellen koneita ja kokeillaan, miten osaan noudattaa minulle annettuja koodeja eli ohjeita.

Esikoululaiset ovat ihania, sillä he ovat aina valmiita kokeilemaan uusia asioita, leikkimään ja heittäytymään. Uskon, että pelkästään ajatus roboteista ja niiden leikkimisestä innostaa lapsia. Lisäksi kaikilla eskari-ikäisillä on jo kokemusta erilaisista laitteista ja sovelluksista ja usein ne ovat lapsista kiehtovia. Tämä seikka lisännee lasten mielenkiintoa, intoa ja motivaatiota työskentelyyn.

Yhteistyö on oppituntikokonaisuudessa keskiössä. Suurin osa tehtävistä on ryhmä- ja paritehtäviä. Aikuinen on mukana ryhmien toiminnassa ihmettelijän roolissa sekä toki kokonaisuudessa myös opastamassa, ohjaamassa ja suuntaamassa toimintaa.

1. Kapteeni käskee

Ensimmäinen leikki on perinteinen Kapteeni käskee. Ensin aikuinen on kapteeni ja antaa ryhmälle ohjeita. Leikin ideana on, että kapteenin ohjetta tulee noudattaa vain silloin, kun hän sanoo: "kapteeni käskee". Kapteeni voi yrittää jekuttaa ryhmää antamalla ohjeita ilman sovittua muotoa. Opettajan jälkeen myös lapset saavat olla vuorollaan pienryhmissä kapteeneja ja antaa toimintakäskyjä.

Leikin pedagogisena pointtina on pohtia ihmisen ja koneen vuorovaikutusta. Me ihmiset voimme ymmärtää epäselviä tai epäsäännönmukaisia viestejä sekä tulkita niitä, mutta tietokone toimii vain täsmällisten käskyjen avulla. Leikin kautta lapset johdatetaan keskustelemaan, millaisia viestejä koneelle tulee antaa, jotta se toimisi halutulla tavalla.

Välineet: leikkiin ei tarvita erillisiä välineitä

2. Robottileikki konkretia (ope robottina)

Toisessa leikissä oppilaiden tehtävänä on ohjata operobottia yhteisesti sovitussa toiminnassa. Lapset on jaettu 4 oppilaan pienryhmiin ja jokainen oppilasryhmä saa ohjata yhtä luokan aikuista. Lapset saavat valita itse, minkä tehtävän (esim. kukkien kastelu, kahvinkeitto, legorobotin rakentaminen...) he haluavat opastaa. Opettaja on todella tärkeä totella ja toteuttaa vain hyvin selkeitä, oikeasuuntaisia ohjeita.

Leikissä jatketaan tietokoneen ja ihmisen vuorovaikutuksen harjoittelua. Idea leikkiin on saatu Ohjelmoinnin ABC varhaiskasvatukseen -materiaalista.

Välineet: leikkiin ei tarvita erillisiä välineitä

3. Matto ja toimintaohjeet

Kolmantena toiminnallisena harjoituksena on parityöskentely, jossa toinen oppilas vuorollaan johdattaa toisen tietyn kuvion luokse isolla, lattialla olevalla satamatolla. Satamaton muovitaskuihin on laitettu erilaisia kuvia ja symboleita. Lapset käyvät vuorollaan nostamassa pinkasta kortin, josta löytyy matossa oleva kuvio. Lapsen tehtävänä on johdattaa pari sovitusta kohtaa satataulua nostamansa kuvion luokse kulkien joko vaaka- tai pystyrivejä pitkin.

Harjoittelun jälkeen tehtävä vaikeutuu. Oppilaan kortissa onkin päätepiste sekä 3 kuviota, jonka kautta pari tulee ohjata sinne.

Pedagogisena tarkoituksena harjoituksessa on harjoitella konkreettisesti ohjelmoinnin alkeita ja vaiheittaisten toimintaohjeiden antamista sekä seuraamista.

Välineet: lattialle levitettävä, muovitaskullinen satamatto, kuvakortit muovitaskuihin sekä ohjekortit lapsille.

4. Robottileikki symboleina -> tietokoneen kieli

Neljäntenä leikkinä harjoitellaan symbolien tulkitsemista ja sitä kautta toimintaohjeiden noudattamista. Oppilaille esitellään robottien symbolikieli (esim. +, #, %, & yms.). Jokaiselle symbolille annetaan konkreettinen sana. Oppilaiden tehtävänä on tulkita pareittain robotin viestit sekä toteuttaa niihin kätkeytynyt ohje. Lopuksi oppilaat saavat itse muodostaa viestejä robotin symbolikielellä tai innokkaimmat voivat jopa keksiä oman koneen symbolikielen merkkeineen ja niiden "suomennoksineen".

Pedagogisena ajatuksena on viedä koodaamisen ajatusta abstraktimpaan suuntaan pohjustamaan koneen kielen erilaisuutta ihmisen kieleen verrattuna.

Välineet: etukäteen mietityt symbolit ja niiden tarkoitukset A3-paperilla sekä valmiita viestejä, joita oppilaat voivat kääntää.

5. Tietokoneen kieli

Viimeisessä tehtävässä on värityskuvia, jotka ovat jaettu ruutuihin. Ruuduissa lukee joko 0 tai 1. Värittämällä ohjeen mukaan ykkösiä tai nollia saa lapsi esiin erilaisia kuvioita.

Pedagoginen ajatus on opettaa lapselle tietokoneen kieltä. Nyt lapsi voi konkreettisesti tehtävässä jutella tietokoneelle oikealla kielellä. Lapselle voi vaikka kertoa, että hän on nyt tietokone ja kun hän suorittaa oikeaa koodia (värittää oikeita ruutuja) viestii hän printterille, millainen kuva pitää tulostaa. idea harjoitukseen on Lumatikka 2 -kurssilta.

Välineet: valmiit värityskuvat ja värikynät

Lähteet:

Lumatikka luennot

Ohjelmoinnin ABC varhaiskasvatukseen

Esiopetuksen opetussuunnitelma

Espoon esiopetuksen opetussuunnitelma

Algoritmileikkejä, esiopetus

2.11.2021

Suunnitelmani koostuu eskaritoiminnasta kahtena aamupäivänä: tällöin opetellaan sekä Nestorin Neliökieli että Amana Taputtajan Läps-Taps-kielet. Tämän jälkeen opettaja heittäytyy Harri-robotiksi ja lopulta tehdään legoista labyrinttirakennelmat, joissa suuntaohjeiden mukaisesti liikutellaan robottiukkoa. Toiminnan jaksaisin niin, että ensimmäisenä aamupäivänä keskittyisimme Nestorin ja Amanan symbolikieleen ja siihen liittyvään toimintaan ja seuraavana päivänä pienen kertauksen jälkeen lapset pääsevät ohjaamaan Harri-robottia, jonka jälkeen tehdään labyrintit. Eskareiden innostuksesta riippuen toimintaa voidaan jatkaa pidempäänkin.

Aihepiirin valinta ja rajaus

- missä oppiaineessa ja mihin aihepiiriin haluan soveltaa ohjelmointia ja miksi: varhaiskasvatuksessa ei noudateta kouluille tyypillistä oppiainetoimintaa, joten valitsen monilukutaidon. On tärkeää, että eskarit oppivat tavallisen lukemisen alkeistaitojen (mm. kirjaimet ja äänteet) ohessa symbolien ”lukemista” ja oppivat samalla leikin ja toiminnallisuuden kautta perusteita ohjelmoinnista.

- Minkälaisia ilmiöitä valitsemaani aihepiiriin liittyy: Lukemisen opettelu on monella eskarilla toiveena eskarivuodelle. Samalla kun lapset opettelevat lukemisen taitoja (kirjaimet ja äänteet), he voivat opetella tunnistamaan erilaisia symboleita ja niiden merkityksiä. Tämä on eskareiden kohdalla luontaista toimintaa, koska suurin osa ei tunnista vielä kaikkia kirjaimia eikä osaa vielä lukea. Näin lukemiseen käytettävät kirjain-symbolit tulevat ”kautta rantain” tutuiksi kuvasymbolien kautta eli ne saavat merkityksen. Lapset ovat pohtineet sääntöjen merkityksiä ja syitä niiden olemassaoloon. Lisäksi rakenteluleikit erityisesti legoilla on ollut mieluisaa ja Minecraft-peli on ollut monien ”huulilla”.

- Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy: symbolit ja niiden merkitykset, niihin tutustuminen ja niiden käyttäminen, keskittymisen taidot, yhdessä opettelu ja tekeminen, leikillisyys ja hauskuus, ohjelmointiin tutustuminen leikin kautta: ohjeiden antaminen, ongelmanratkaisutaitojen harjoittelu, hienomotoriset ja karkeamotoriset taidot, suunta-käsitteet (eteen, taakse, oikea, vasen, stop).

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

- millaisia oppimistavoitteita asetat oppimiskokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen: ohjelmointiin liittyvän motivaation herättely, keskittyminen toimintaan, yhteistyön taidot, kaverin auttaminen ja kannustaminen, tutustuminen symboleiden käyttämiseen, motoriset taidot, suunnat ja sijainnit, viestinnän merkitys, viestintävälineet, tunteet liittyen lapsen omaan oppimistilanteeseen, ohjeiden täsmällinen antaminen, looginen ajattelu

- Miten opettaja tai joku muu arvioi opiskelijan osaamista? Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään: Opettajana arvioin yllä olevien tavoitteiden toteutumista sekä yksilö- että yhteisötasolla. Tärkeintä olisi selvittää, mitä lapsi oli oppinut symboleiden käytöstä, ohjeiden antamisesta ja sijainneista. Tärkeää samalla selvittää, kuinka lapsi selvisi yhteistyön hetkistä ja itsekseen tehtävistä toiminnan osista.

- Miten oppilas arvioi omaa osaamistaan: Voin esimerkiksi selvittää lapsilta, mitä he mielestään oppivat, he voivat vaikka kertoa omin sanoin, mitä he ovat oppineet. Tähän on mahdollista käyttää esim piirtämistä, jonka avulla lapsi piirtää jonkun asian, joka jäi toiminnasta mieleen ja muutamilla lisäkysymyksillä lapsi kertoo mikä oli mukavinta ja merkityksellisintä.

Työskentelyvälineet ja opetusmenetelmät

- mitä välineitä oppimiskokonaisuudessa käytetään: Ohjelmoinnin ABC varhaiskasvatukseen -vihko, kuvasymbolit, hahmot Nestori ja Amana, legot, labyrintti-tuloste (pareille yksi ja yksi yhteinen), nuolet eteen- ja taaksepäin sekä käänny oikealle, käänny vasemmalle, stop-merkki, kahvikuppi

- Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan: mitään pitkää selitystä ei näinkin toiminnalliseen tarvita ryhmässäni, riittää kun Nestori ja neliökielen symbolit esitellään aluksi ja sitten lähdetään toimimaan. Alussa kertaus on oleellista jotta toiminta tulee tutuksi, ennen kuin siirrytään seuraavaan osioon. Labyrintin rakentamisessa legoilla keskustellaan labyrinteistä ja kokeilun kautta saadaan varmasti kehitettyä jo enemmän kolmiulotteisuutta niihin.

- Millä perusteilla oppilaat motivoituvat tästä aiheesta: meillä lähdetään mielellään mukaan kaikkeen uuteen. Toisaalta, viime vuonna samassa ryhmässä olleet ehkä muistavatkin opetelleensa Nestorin neliökieltä, joten aloitus varmasti lähtee liikkeelle hyvin. Motivaatio pysyy yllä kun hiljalleen lisätään vaikeusastetta mutta pidetään huoli myös siitä että toiminta on hauskaa ja riittävän helppoa. Legojen rakentaminen on meillä mieluisaa eli erityistä motivointia ei tarvita. Pieni käsitteisiin tutustuminen on myös paikallaan eli lapsille voi jo puhua ohjelmoinnista ja koodauksesta ja että nyt opetellaan tutustumaan niihin, jotta voidaan jutella lisää tietoteknisistä laitteista ja niiden toiminnasta.

- Miten opetuksessa rohkaistaan yhteistyötä: opettajana minun täytyy huomioida jokaisen motivaatio ja sitoutuminen sekä onnistuminen ja vinkata kaikkia seuraamaan toisiaan, koska neliökielen ratkaisun on kaikkien toiminnasta kiinni ja että esim nuolten kanssa kaikkien täytyy liikkua samaan suuntaan. Lisäksi on tärkeää pitää huolta, että lapset saavat keskustella ja neuvoa tosiaan toiminnan aikana. Labyrintin tekemisessä on tärkeä varmistaa, että tehdään ainakin yksi ensin

yhdessä ja kun parin kanssa jatketaan, että parit muodostuvat sekä osaajista että harjoittelijoista.

- Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelua: käytän ongelmanratkaisua runsaasti päivän aikana eli aina kun lapsi pohtii jotain, hänellä on ongelma, johon hän etsii ratkaisua. Tätä taitoa varmasti tullaan vahvistamaan suunnitelmassani toiminnassa. Lisäksi ryhmässäni on lapsia, joilla on toiminnan ohjauksen ja itsesäätelyn haasteita. Täten toiminnan aikana kokeiltua voi yhdistää näihin haasteen paikkoihin: miten ohjelmoisin itseni toimimaan niin, että annan itselleni käskyjä osissa (otan ulkovaatteet esiin, puen ensin housut, puen seuraavaksi takin jne).

Ohjeiden ja käskyjen harjoittelua, esiopetus

27.11.2021

Esiopetukseen tuntisuunnitelma, jossa tutustutaan ja harjoitellaan ohjeiden sekä käskyjen antamista ja suorittamista

Kohderyhmä ja kesto: esiopetusikäiset. Toiminta tapahtuu pienryhmissä (n. 7-11 lasta) tunnin pituisina tuokioina kahtena erillisenä päivänä. Lapset jaettu 2-3 hengen ryhmiin.

Välineistö: Logico-pelikehykset ja Kokeile koodaamista -kortit, lautapelejä (esim. Muuttuva labyrintti, Robogem), 2kpl BeeBottia

Lähtökohta suunnittelulle: Valtakunnallinen sekä Helsingin esiopetussuunnitelma sisältää ohjelmointiin tutustumisen esiopetusvuoden aikana. Olen laatinut toteutuksen siten, että tämä suunnitelma sisältää aloituksen eli aiheeseen tutustumisen ja sen jälkeen asiaa käsitellään lisää lasten kiinnostuksen pohjalta. Suunnitelmassa on huomioitu lasten nykyinen tietotaso aiheesta sekä muut opeteltavat asiat ja kokonaisuus on tarkoitus toteuttaa esiopetusryhmässäni keväällä.

Aihepiiri ja oppimistavoitteet: Kahden tunnin aikana harjoitellaan ohjeiden ja käskyjen antamista ja niiden toteuttamista. Tavoitteena on toimintojen sanoittaminen ja selkeiden, mahdollisimman yksiselitteisten ohjeiden antaminen. Tarkoitus on myös herättää kiinnostus koodaamiseen ja teknisten laitteiden toimintaan sekä ongelmanratkaisuun ja loogiseen ajatteluun. Lisäksi harjoitellaan ilmaisua, kuuntelemista ja ryhmätyötaitoja.

Arviointi: Lapset arvioivat toimintaa peukuttamalla ja omaa oppimista kolmiportaisella asteikolla (LUMATIKKA 3: Ymmärrystä ongelmanratkaisuun). Arvioinnin portaat: I taso: Osaan noudattaa ohjeita, II taso: Osaan antaa selkeitä ohjeita, III taso: Osaan toimia ryhmässä.

Opettaja seuraa lasten kiinnostuneisuutta, oppimista ja ryhmätyöskentelyä, minkä pohjalta hän voi suunnitella jatkoa tälle kokonaisuudelle.

Toteutus:

Ennakkosuunnittelu: välineistön kokoaminen, pienryhmien/parien muodostaminen, arviointilomakkeen ja kotitehtävän valmistelu, tiimin muiden aikuisten ohjeistaminen

<u>Oppitunti 1</u>

Toteutus pistetyöskentelynä, jossa kolme erilaista toimintaa: Logico-pisteellä ratkaistaan koodaustehtäviä, pelipisteellä pelataan loogista ajattelua kehittäviä pelejä, robottipisteellä koodataan Beebotteja.

Aloitus. Yhdessä piirissä pohditaan ohjeiden ja sääntöjen tarkoitusta, noudattamista ja hyötyä. Lapsia pyydetään kiinnittämään huomiota pisteillä tarvittuihin taitoihin ajatellen tehtävissä onnistumista. Käydään läpi pisteiden sisältö ja muistutetaan yhteistyötaidoista ja kaverin auttamisesta. Opettaja näyttää myös, miten BeeBot toimii.

Toiminta. Lapset kiertävät pisteillä 2-3 hengen ryhmissä. Ensin suoritetaan Logico- ja pelipisteet, minkä jälkeen pääsee koodaamaan BeeBottia. Koodauspisteellä aikuinen auttaa laitteen käytössä. Jokaiselle pisteelle on varattu n.20min.

Koonti. Loppupiirissä keskustellaan pisteiden tehtävistä. Käydään uudelleen läpi aloituksen pohdintoja ja lapset kertovat huomioistaan liittyen tehtävissä onnistumiseen. Opettajan kysymyksinä esim. ”Ymmärsitkö ohjeet ja toimit niiden mukaisesti? Osasitko käskyttää BeeBottia? Pyysitkö kaverin apua?” . Seuraavaa kertaa varten annetaan kotitehtävä, jossa lapsi miettii vanhempien kanssa jonkin kotona olevan laitteen toimintaa, piirtää laitteen ja laatii käyttöohjeet sille. Palautus ennen tuntia 2.

Oppitunti 2

Opettaja käy ennen tuntia läpi palautetut kotitehtävät ja suunnittelee niiden pohjalta jälkimmäisen tunnin mm. kerää materiaalia ja apuvälineitä esityksiä varten. Ennen tuntia pystytetään sermi opetustilaan.

Aloitus. Opettaja esittelee piirin ohjelman. Lapset pääsevät pareittain esittämään robottia sekä robotin käyttäjää. Tarkoituksena on saada robotti käyttämään laitetta, jonka lapsi on kotitehtävään valinnut. Sermin eri puolille asettuvat robotti ja ohjaaja. Ohjaaja ei saa paljastaa valitsemaansa laitetta. Robotti pyrkii toimimaan vain kuulemiensa ohjeiden mukaan ja katsojat yrittävät arvata, mistä laitteesta on kyse. Käydään läpi itsearviointilomake.

Toiminta. Edellisen kerran parit/ryhmät esiintyvät vuoron perään. Opettaja tukee esityksiä ohjeistavin kysymyksin. Ennen koontia lapset arvioivat arviointilomakkeella omaa toimintaa tuntien aikana.

Koonti. Lopuksi pohditaan, minkälaista oli toimia eri rooleissa ja mikä niissä oli helppoa/vaikeaa. Laaditaan ohjeet hyvien käskyjen antamisesta.

Jatkotoiminta: Opettaja arvioi tuntien onnistumista itsearviointien ja ryhmän aikuisten havaintojen pohjalta. Lasten kiinnostuksen ja osaamisen perusteella käskytoimintojen harjoittelua voidaan jatkaa esimerkiksi sääntöleikeissä, liikunnassa, askartelussa. Kun asia alkaa sujua, mukaan voidaan ottaa tietokonepelejä tai ScratchJr:lla koodausta.

Lähteet: Suunnitelma on itse kehitelty ja lähteitä ei ole varsinaisesti käytetty. Vinkkejä ja ideoita on saatu Jenni Vartiaisen Ohjelmoinnin ABC varhaiskasvatuksessa -julkaisusta, tämän kurssin varhaiskasvatukseen liittyvistä linkeistä, LUMATIKKA-kursseista Ymmärrystä ongelmanratkaisuun ja Varhaiskasvatuksen ja esiopetuksen matikkaa lapsilähtöisesti.

Ongelmia ja ohjelmointia, esiopetus

Tiia Leppänen, CC BY-SA 4.0

10.4.2022

Ongelmia ja ohjelmointia

Kesto: 1 oppitunti
Välineet: Lattiateippaus tai muu vastaava tapa toteuttaa ohjelmointiruudukko, jossa liikutaan. Esikoululaisen arkeen liittyviä toimintokuvauksia/ongelmia kirjallisesti ja suullisesti sekä näihin linkittyviä välineitä/teknologioita.
Aikaisemmat tiedot: Varsinaista aikaisempaa perehtyneisyyttä ei tarvita, joskin tunnin alussa on hyvä hieman pohjustaa, miksi tämä harjoite tehdään
Tavoitteet: harjoittaa lasten ongelmanratkaisutaitoa, yhteistyötaitoja sekä ohjelmointikielen hallintaa. Tukee algoritmisen ajattelun kehittymistä tarjoten kehollisen kokemuksen komentosarjan tuottamasta suotuisasta tuloksesta. Harjoitus linkittyy esikoulun sisällöistä eritoten kielen rikas maailma (ohjeiden antaminen, kommunikointitaidot) ja tutkin ja toimin ympäristössäni (teknologiat arjessamme, algoritminen ajattelu) teemoihin.
Valmistelu: ruudukot oppilasmäärän mukaan (voidaan toteuttaa myös pienryhmissä, jolloin toiminto tapahtuu kahtena erillisenä oppituntina) sekä tarvittavat tavarat ruudukkoon eri paikkoihin (osa voi olla myös samassa kohtaa, jolloin lapsen pitää pystyä valitsemaan oikea tavara). Kuva/tekstikortit esikoululaisen arjen tilanteista. Voi käyttää myös tablettia, jossa valmiina kuvasarja, jota selataan eteenpäin ongelman ratkettua.

Tunnin kulku:
Kokoonnutaan yhteen ohjelmointiruudukkojen luokse. Ruudukkoja on jokaiselle parille TAI kolmikolle oma. Ruudukkojen ei tarvitse olla mahdottoman isoja, varsinkin alaspäin eriyttäessä ruudukko voi olla hyvinkin maltillinen jollakin parilla, mikä helpottaa hahmottamista ja työskentelyä parin ohjauksessa. Ruudukon toteutus voi olla teippauksilla, muotopaloilla tms.
Opettaja kertoo omin sanoin, että tänään harjoitellaan ongelmien ratkaisua, jossa apuvälineenä tarvitaan omaa luokkakaveria. Opettaja kuvailee, kuinka hänellä on kädessään lappuja, joissa on kirjoitettu/kuvattu eskarilaisille hyvinkin tuttuja erilaisia tekemisiä. Tekemiset voivat olla esimerkiksi piirtäminen, syöminen, muovailu, kaupassa käyminen, hammaspesu tms. Lukeville oppilaille voidaan antaa vain tekstillinen lappu ja muille kuvakortti, joka osoittaa tekemisen. Vaihtoehtoisesti ei-lukeville voidaan myös tekeminen kertoa suullisesti.
Opettaja kertoo, kuinka lasten tehtävänä on keksiä, mitä tavaroita tai teknologisia keksintöjä tarvitaan kyseisen tekemisen toteuttamiseen (ongelmanratkaisu, mitä tarvitsen piirtämiseen, entä kaupassakäymiseen). Lasten tulee ruudukossa ohjata oma pari juuri oikeiden tavaroiden luokse ja kerätä nämä mukaansa. Opettaja havainnollistaa asiaa ruudukon äärellä, tekemällä itse muutaman esimerkin. Opettaja samalla korostaa, kuinka tärkeää selkeät ja tarkat ohjeet ovat sekä niiden noudattaminen ja yhteistyön tekeminen.
Ohjeiden jälkeen jaetaan lapset pareihin/kolmikkoihin sekä työskentelypisteisiin (muista mahdollinen eriyttäminen!). Pareille/kolmikoille jaetaan tekemistä kuvaavat laput, esimerkiksi 2-3 lappua kerrallaan. Lapset sopivat kuka on ensin

ohjelmoitavana ja kuka jakaa ohjeita. Tässä harjoitellaan myös vuorottelua. Haettuaan kaikki esineet, tuodaan saalis ja laput näytettäväksi opettajalle. Kerrotaan, mikä tekeminen meillä tässä lapussa oli ja mitä välineitä siis tarvittiin sen suorittamiseen ja miksi (syy-seurassuhteet, teknologisten ratkaisujen hahmottaminen osana arkista toimimista. HUOM! teknologiaa myös nykyään itsestään selvät innovaatiot, kuten nykyaikaiset hammasharjat ja syömävälineet). Opettajan tehtävä on rikastuttaa oppilaiden ajatusmaailmaa muutamalla kysymyksellä, MIKSI nämä asiat tarvittiin ja MIKSIKÖHÄN nämä ovatkaan aikanaan keksitty? Jotta.....? Tärkeää on kuitenkin ottaa huomioon aika, älä anna ryhmien liian kauan odottaa jonossa, joskin pieni odottaminen on myös hyvää harjoitusta, mikäli muutama ryhmä tulee yhtä aikaa paikalle. Tarkistuksen ja keskustelun jälkeen, annettaan muutama lappu lisää ja muistutetaan esimerkiksi jostain tärkeästä asiasta, johon kiinnittää tällä kierroksella enemmän huomiota (ohjeiden anto, yhteistyö ja yhdessä työskentely tms).
Opettajan tehtävänä on havainnoida lasten työskentelyä sekä kysyä myös heiltä tarkistuspisteellä omia arvioita, kuinka tavaroiden etsiminen sujui ja oliko joku hankalaa (itsearviointi, yhteisarviointi). Tämän perusteella toimintaa pyritään hetkessä parantamaan.
Toiminnon lopuksi kokoonnutaan vielä yhteen ja opettaja tiivistää lyhyesti vielä tunnin kulun havaintojensa pohjalta ja tarttuu sellaisiin aiheisiin, jotka vielä jäivät mietityttämään tai vaikuttivat lapsista hankalalta. Huumoria voidaan ottaa vielä loppuun mukaan ja opettaja voi itse asettua ruudukolle ja pyytää jotakuta lasta antamaan ohjeita niin opekin hieman kokeilee tavaroiden keräämistä. Opettaja voi tahalleen toimia hassusti, joka tietysti lapsista hauskaa, kun ope munailee! Tässä vielä konkreettisesti näytetään, kuinka oleellista on, että toimintakäskyt ymmärretään ja annetaan tarkasti. Voidaan viimeisenä vielä kysyä, mitkä asiat voivat vaikuttaa siihen, että ope tai joku muu ymmärtää ohjeenväärin (esim. unohdetaan sanoa jotain, entä jos onkin kuuro/sokea -> mitä sitten pitäisi tehdä).
Ohjelmoidaan oppilaat jatkamaan eteiseen pukemaan ulos korostetusti koodikielellä (Nouse ja kävele eteiseen, pue vaatteesi ja kävele ulos, LEIKI!)

Jatko:
Lähipäivät ja viikot voidaan ohjelmointikieltä käyttää korostetusti esimerkiksi siirtymätilanteissa tai tehtävien ohjeistuksessa, jolloin oppilaiden ajatukset palaavat tähän hetkeen uudelleen ja uudelleen. Siirtymät ovat esikoulun arjessa ohjeiden antamisen ohella yksi selkeimmistä hetkistä, kun toimintayksikön toivotaan ja odotetaan toimivan juuri annetun ohjeen mukaisesti :) Joskin tietysti suotavaa on jatkaa vielä aiheesta aikuisjohtoisesti toiminnon avulla myös lisää.

Ohjauksesta:
Kannusta yhteistyöhön ja toiset huomioivaan työskentelyyn (äänenvoimakkuuden taso, kuuluva mutta ei häiritsevä huutava. Kannustus, turhauman sieto ja kohteliaisuus. Vuorottelu ja rohkeus sanoa oma mielipide siitä, mitä tavaroita tarvittaisiin)

Panosta selkeyteen ja havainnollisuuteen alun esimerkissäsi ja puheessasi
Muista myös vivahteikkuus kommunikoinnissa sekä oma aito kiinnostus lasten ajatuksia kohtaan
Heittäydy lopun huumorointiin!
Eriytä tarpeen mukaan, jos et tiedä lähtötasoa, aloita helposta! -> kaikille onnistumiset taattu!

Ideaa ammennettu tämän tehtäväosion teksteistä sekä erityisesti materiaalista: kesäleiri-2018-materiaali.pdf (innokas.fi)

Kirjainten kertaaminen BeeBot-robottien avulla, esiopetus

Nimetön,

22.9.2022

Tuntisuunnitelma
Esiopetus. Bee Bot -robotin ohjelmointi ja opittujen kirjainten kertaaminen

Aihepiirin valinta ja rajaus

Suunnitelmani liittyy esiopetuksen Kielten rikas maailma sekä tutkin ja toimin ympäristössäni -oppimisen alueisiin. Tarkoituksena on ohjelmoida Bee Bot-robottia löytämään oikeat kirjaimet. Tehtävää tehdään ryhmätyönä, mutta yksittäin.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Ryhmässä on tutustuttu tehtävässä oleviin kirjaimiin, joten niitä kerrataan ja tehtävää suoritettaessa voikin arvioida ovatko oppilaat oppineet tunnistamaan ja nimeämään kirjaimet. Ohjelmointia on harjoiteltu nuolikorteilla (mm. Robocem- lautapeli) ja Bee Bot-robottiin oppilaat ovat aikaisemmin jonkin verran tutustuneet. Käytössä ovat olleet käskyt liikkua eteenpäin ja kääntyä oikealle tai vasemmalle. Oppilaat pystyvät arvioimaan omaa ohjelmoinnin osaamistaan työskentelyn aikana seuratessaan, miten robotti liikkuu. He myös pystyvät muuttamaan ohjelmointia, jolloin opettaja voi arvioida ongelman ratkaisu- ja yhteistyötaitoja virhettä etsittäessä.

Työskentelyvälineet ja opetusmenetelmät

Tarvittavat välineet: Bee Bot-robotti (tai vastaava esim. Robot Mouse) ja nuolikortteja ja vihreä kortti jokaiselle ryhmälle. Isoja kirjainkortteja ohjelmointimattoon tai isolle paperille ruudukkoon piirretyt kirjaimet. Isoja nuolikortteja ja vihreä kortti. Tehtäväpaperi, jossa ruudukko jokaiselle oppilaalle. Bee Bot -robotti toimii itsessään motivoivana elementtinä. Opettaja voi myös miettiä ryhmien kokoonpanoja motivoinnin ja oppimisen näkökulmasta.

Johdanto

Pulpetit on siirretty luokkatilan seinien viereen, jotta keskelle lattiaa jää mahdollisimman paljon tyhjää tilaa. Opettaja asettaa taululle isoja nuolikortteja. Oppilaat menevät seisomaan takaseinän viereen riviin. Opettaja asettaa vihreän kortin ohjelman loppuun käynnistääkseen ohjelman. Yhdessä lähdetään liikkumaan nuolien osoittaman reitti. Sivulle päin osoittavan nuolen kohdalla käännytään, ei liikuta (aivan kuten Bee Botkin tekee). Keskustellaan siitä, miksi päädyttiin tiettyihin paikkoihin. Tämän jälkeen jokainen oppilas saa vapaasti valita lähtöpaikkansa. Opettaja tekee uuden ohjelman nuolikorteista taululle ja näyttää yhtä kuvaa kerrallaan ja oppilaat liikkuvat nuolien mukaan. Keskustellaan siitä, miksi oppilaat nyt päätyivät eri paikkoihin. Miten oppilaat olivat tulkinneet nuolet (liikkuivatko ylöspäin osoittavan nuolen kohdalla kohti taulu vai etenivätkö kulkusuunnassa)? Keskustellaan miten tulisi liikkua ja miksi.

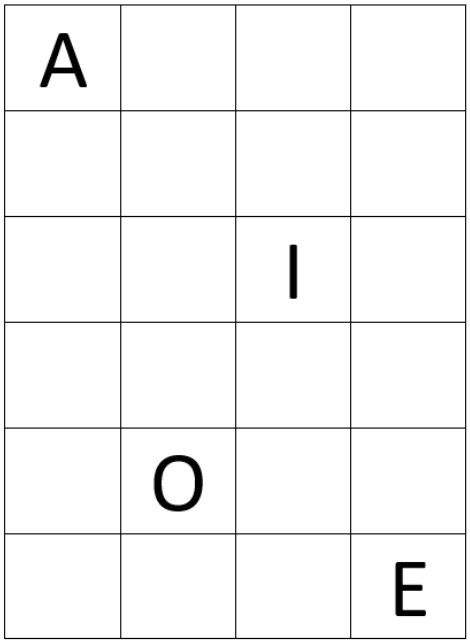
Toiminta

Tämän jälkeen oppilaat saavat ottaa sivupöydältä ohjelmointimaton tai ruudutetun paperin, jossa on kirjaimia. Oppilaiden tehtävänä on vuorotellen ohjelmoida Bee Bot löytämään haluttu kirjain. Ryhmästä yksi oppilas saa asettaa Bee Botin valitsemaansa lähtöruutuun, jonkin kirjaimen päälle. Bee Bottia ei saa kääntää, eikä siirtää. Toinen oppilas kertoo kirjaimen, jonka luokse Bee Botin tulisi päästä. Kirjain ei saa olla sama, jonka päällä robotti jo on. Ohjelmoija kirjoittaa nämä kirjaimet paperiinsa. Tämän jälkeen hän suunnittelee reitin nuolikorteilla. Suunnitelman valmistuttua se ohjelmoidaan robottiin ja käynnistetään ohjelma. Koko ryhmän tehtävänä on seurata robottia. Jos robotti päätyy oikeaan kohtaan oppilas kirjoittaa reitin paperiinsa. Jos robotti päätyy väärään paikkaan, mietitään yhdessä, miten ohjelmaa tulisi muuttaa. (Bee Bot tulee muistaa tyhjentää edellisestä ohjelmasta ennen uudelleen ohjelmointi.) Vaihdetaan rooleja.

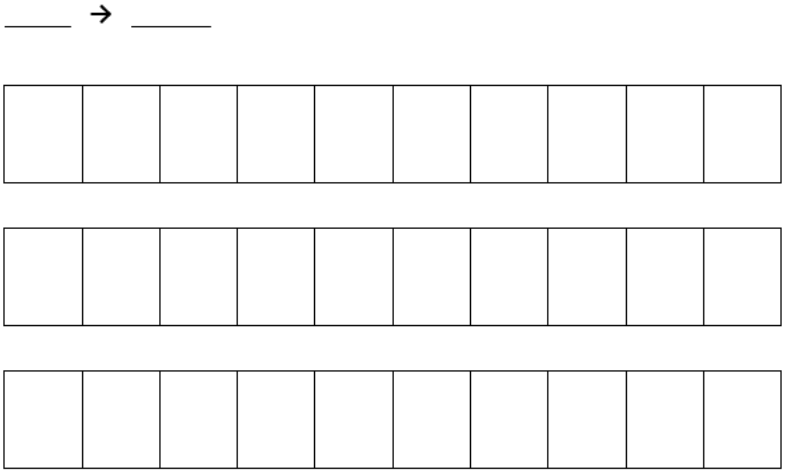
Yhteenveto/ arviointi

Keskustellaan siitä, miten tehtävä sujui. Mikä oli helppoa ja missä tarvitaan vielä harjoittelua? Jakaudutaan ryhmiin sen mukaan, minkä reitin oppilas oli ohjelmoinut. Keskustellaan, että löytyykö kaikkia mahdollisia yhdistelmiä? Oppilaat saavat verrata ohjelmian keskenään. Ovatko ne samanlaisia? Miksi eivät? Keskustellaan miten samaan lopputulokseen voi päästä monella eri tavalla. Lopuksi mietitään missä muualla on ohjelmointia käytetty (esimerkiksi Kapteeni käskee ja värileikki). Missä sitä voisi käyttää?

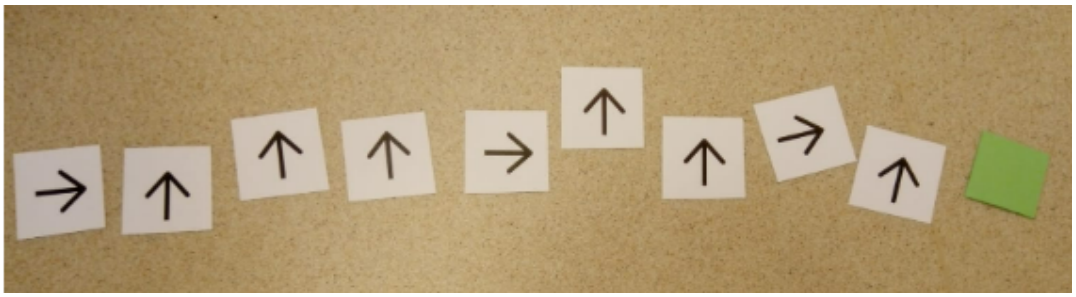
Esimerkki ruudutetusta paperista, jossa neljän kirjaimen harjoitus. Ruudun koko noin 15x15 cm

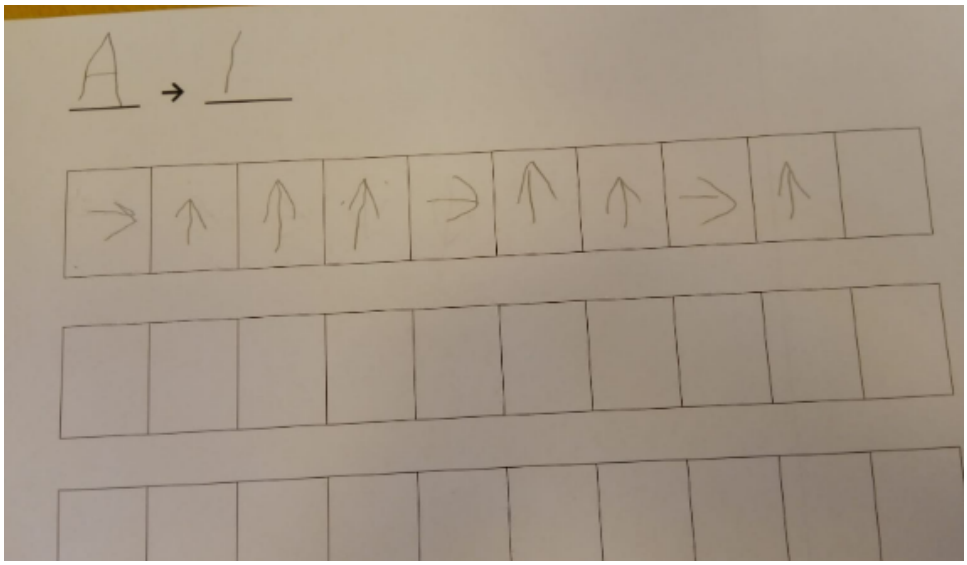


Oppilaan lomake. Ylimmälle riville kirjoitetaan mistä kirjaimesta mihin ohjelma tehdään. Ruutuihin piirretään toimintaohje nuolilla.



Yhden oppilaan ratkaisu





Koodausta ekalle luokalle ilman koneita ja esimerkkinä ohjeiden anto, 1.lk

Salla Liusjärvi, CC BY-SA 4.0

5.5.2019

Koodausta ekalle luokalle ilman koneita ja esimerkkinä ohjeiden anto

Aihepiirin valinta ja rajaus

Toimin laaja-alaisena erityisopettajana ja erityinen kiinnostuksen kohteeni on alkuopetus. On siten luontevaa, että valitsen alkuopetuksen kohteekseni. Lisäksi olen itse hyvin aloitteleva koodaaja, joten perustehtävien opetuksesta on hyvä aloittaa. Oppiaine on matematiikka, johon koodaus sopii oikein hyvin alkuopetuksessa. Koodausta harjoitellaan ilman koneita, tarkoitus on kehittää algoritmista ajattelua ja ohjelmoinnin perusteita, mikä pohjustaa jatkoa.

Valitsemaani aihepiiriin eli matikkaan ja alkuopetukseen liittyy monia ilmiöitä. Ensinnäkin koodaus soveltuu moneen oppiaineeseen ja tukee monen alkuopetuksen oppiaineen tavoitetta. Toisaalta taas oppilaiden lähtötaso on hyvin vaihteleva ja matematiikan osaamisessa on monen vuoden ero vaikka ykkösluokkalaisten kesken.

Tavoitteet on OPS:sta esim. alkuopetuksen matematiikan tavoite T4 soveltuu tähän aihepiiriin: Ohjata oppilasta kehittämään päättely- ja ongelmanratkaisutaitoja. Lisäksi toki OPS:ssa sanotaan, että alkuopetuksessa pitää käsitellä ohjelman toiminnan ja testauksen alkeita sekä lasten pitää saada ja jakaa keskenään ikäkaudelle sopivia ohjelmointitehtäviä.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Koska ollaan aivan alkuopetusessa, tavoitteet on edellä mainittujen lisäksi tukea kiinnostusta matematiikkaan, kielentää matikkaa sekä opettaa matikkaa ja ohjelmointia vuorovaikutuksessa toiminnallisesti muiden kanssa. Idea on, että oppilaat alkavat ymmärtää ohjelmoinnin perusteita, jonka päälle on helpompi rakentaa koodausta myös eri kielillä ja laitteella jatkossa. Oppilaat innostuvat ja heidän algoritminen ajattelunsa jatkaa kehittymistään.

Arviointia tehdään alkuopetuksen oppilailla seuraavasti. He saavat arvioida omaa osaamistaan kuvallisen viisikentän avulla. Eli he arvioivat jokaisen tehtävän sujumisen yksin tai yhdessä (riippuen siitä, onko tehtävä tehty yksin vai pareittain tai ryhmässä) kuvallisesti viisikentällä (kivaa ja helppoa / kivaa ja tylsää / vaikeaa ja helppoa / vaikeaa ja tylsää / haluan oppia tästä

lisää) opettajan kanssa. Ohjelmointi on tässä tapauksessa kokonaisuus kohdeoppimisen eli matematiikan kanssa ja näitä ei oikein voi tässä vaiheessa vielä arvioida erikseen, koska ollaan aivan alkeissa. Koodausta kun on tarkoitus harjoitella ilman koneita.

Työskentelyvälineet ja opetusmenetelmät

Tehtävät tulevat tältä sivustolta https://csunplugged.org/en/ ja tehtävästä riippuen pärjätään ilman konetta, osa tehdään toiminnallisesti liikkuen, osassa tarvitaan kynää ja paperia, osassa taas erilaisia palikoita ja muita välineitä tai näiden kaikkien yhdistelmä. Oppilaille annetaan jokin tehtävänanto tai haaste, joka oppilaiden pitää ratkaista. Ratkaisua varten tarvitsee ottaa huomioon muutama sääntö, jotka ohjaavat ratkaisua. Idea on, että oppilaat keksivät ja oivaltavat itse ja saavat onnistumisen kokemuksia. Oppilaille samalla selitetään, että kyse on myös tästä tehtävästä että laajemmasta tavasta ajatella ongelmia, koodauksen alkeista. Jokainen tehtävät ohjeistetaan lisäksi antamalla juuri sen tehtävän ohjeet.

Tarkoitus on, että nämä tehtävät eivät ole vain yksittäinen kokonaisuus vaan näitä tehtäviä tehdään joka viikko, muun opetuksen ohessa, myös muilla kuin matikan tunneilla. Lisäksi osa näistä tehtävistä voi olla ns. ylöspäin eriyttäviä tehtäviä eli kun oppilas on saanut omat tehtävänsä tehtyä, hän voi alkaa tehdä näitä tehtäviä.

Oppilaat perinteisesti motivoituvat helposti tehtävistä, jotka ovat toiminnallisia, sisältävät omaa oivaltamista ja ratkaisua. Jos joku oppilas ei motivoidu tehtävästä, häntä autetaan alkuun ja hänelle järjestetään onnistumisen kokemuksia näissä tehtävissä, jotta hänelle muodostuu positiivinen minäkuva ja itsetunto liittyen näihin koodaustehtäviin. Yhteistyöhön kannustetaan teettämällä sellaisia tehtäviä, joita pitää ratkaista yhdessä.

Aiempi osaaminen otetaan huomioon siten, että näille oppilaille (jotka osaavat ratkaista heti nämän ongelmanratkaisutehtävät) annetaan vaikeampia tehtäviä tai sitten he voivat alkaa koodata jo Screatch Juniorilla. He voivat myös toimia yhdessä jonkin haastavan tehtävän parissa.

Oppilaiden ongelmanratkaisutaitojen kehitystä voidaan vielä tuoda esiin siten, että heitä pyydetään soveltamaan jotain ratkaisumallia ihan uuteen tilanteeseen, esim. vaikka luokassa pitäisi muuttaa jotakin, koulussa pitäisi uudistaa jotakin.

Tässä kuitenkin opettajalle yhden tai kahden oppitunnin ohjeistus:

Aihe: Ohjeiden antaminen 1.luokka. Tehtävä on otettu tältä sivulta. https://csunplugged.org/en/topics/kidbots/unit-plan/rescue-mission/#rescue-mission Sivustolta löytyy myös tulostettavat nuolikortit ja muuta materiaalia, sekä kuvalliset ohjeet.

Tavoitteena on, että oppilas ymmärtää, miten liikkua ruudukossa, hän osaa selvittää ohjelman ongelmia ja keskustella niistä, oppilas ymmärtää ja osaa antaa ja toteuttaa tarkkoja ohjeita sekä korjata ohjelman ongelmat.

Paikka: iso ruudukko, joka on piirretty kentälle tai luokkaan.

Opettaja pyytää kaksi vapaaehtoista ja antaa heille ja itselleen seuraavat roolit.

Rooli 1: koodari (opettaja aluksi)

Rooli 2: testaaja (ohjaa Bottia ja etsii virheitä)

Rooli 3: Bot

Ope sanoo: "Olen koodaaja, mutta tarvitsen apuanne. Ohjelmoimme Bottia, mutta emme kauko-ohjaa vaan meidän pitää antaa Botille kaikki ohjeet ennen kuin Botti voi seurata ohjeita. Meidän tarvitsee antaa selvät ohjeet Botille, joka on oppilas XXX. Oppilas YYY on testaaja ja antaa ohjeet Botille. Testaajan tehtävä on etsiä ohjelman virheitä eli bugeja. Ensiksi meidän pitää päättää, mitä ohjelmointikieltä käytämme. Olen valinnut nuolet, jotka kuvaavat eteenpäin menoa ja oikealle ja vasemmalle kääntymistä. Virheiden etsintä on hauskaa, koska ohjelman teon jälkeen voit muuttaa sitä, jotta se toimisi oikein."

Oppilaat oppivat, että yhteen ohjelmaan valitussa kielessä pitää pysyä koko ajan. Kerro Botille, että nuoli tarkoittaa yhtä siirtymistä eteenpäin ruudukossa ja vasemmalle tai oikealle kääntyminen tarkoittaa kääntymistä paikallaan.

Ope sanoo: "Ruudukossa on karkkikulho. Saamme kaikki ottaa yhden karkin, jos osaamme ohjata Botin kulhoruutuun. Botti seisoo ruudukon reunalla.

Ope sanoo: "Nyt aloitetaan, katsotaan kuinka käy. Tässä on ohjeeni testaajalle, anna ne Botille. Tarkkaile, jos ohjeet eivät toimi, etsitään virhe eli bugi ja sitten mietitään, miten virhe korjataan." (huom. ensimmäisen testin ohjeet ovat puutteelliset ja Botti ei pääse karkkikulholle saakka) Kun aloitetaan, nuolet voi laittaa ruudukkoon, jotta oppilaat näkevät, mitä Botin pitää tehdä.

Tässä vaiheessa ope voi kysyä oppilailta, tarvitaanko stop-käskyä ja miksi

ei.

Ope sanoo: “Testaaja, toimiko ohjelma niin kuin halusimme?”

“Nyt korjaamme koodin ja kokeilemme uudelleen. Minkälainen on uusi hyvä koodi?” Uutta koodia kokeillaan samalla tavalla kuin aiemmin. Tämän jälkeen voidaan kokeilla jotain muuta koodia ja katsotaan, mihin Botti päätyy. Ohjelmoija tekee koodin ja testaaja antaa ohjeet Botille ja testaaja etsii virheitä ja Botti liikkuu. Tässä vaiheessa voidaan ottaa uudet vapaaehtoiset.

Mistä tiedämme, milloin Botti on oikeassa paikassa? Silloin kun hän on samassa ruudussa kuin karkkikulho. Pääseekö karkkikulhon luo muita reittejä? Entä miten Botti voi ottaa karkkikulhon mukaansa? Miten hän tulee takaisin lähtöpisteeseen? Minkälainen on hyvä koodi? Muodostakaa kolmen hengen ryhmät, päättäkää roolit ja tulkaa esittämään ratkaisunne isoon ruudukkoon.

Luokassa oppilaat jatkavat kolmen hengen ryhmissä. Heillä on shakkiruudukko. He asettavat haettavan esineen jonnekin ruudukkoon ja hakijan ruudukon reunalle, kasvot kohti keskustaa. Ohjelmoija kirjoittaa paperille nuolilla koodin. Testaaja antaa koodin Botille ja testaaja seuraa, miten Botti selviää tehtävästä. Jos Botti ei pääse haettavan esineen luo, testaaja selvittää koodin bugit ja alleviivaa ne. Tämän jälkeen ohjelmoija kirjoittaa uuden koodin ja antaa sen testaajalle ja koodia testataan.

Ylöspäin eriyttäminen

Esim. shakkiruudukossa on esteitä, eli haettavan esineen luo ei pääse suoraan.

Mitä ajattelua kehittää?

Peräkkäisten ohjeiden keksiminen kehittää algoritmista ajattelua, koska ohjeiden tarkoitus on selvittää jokin tehtävä. Näkevätkö oppilaat jo ennen tehtävän suorittamista lopputuloksen? Osaavatko oppilaat antaa ohjeet? Osaavatko oppilaat tunnistaa harjoituksen jälkeen ohjelmoinnin eri vaiheet, eli koodauksen, testauksen ja bugien etsimisen? Löytävätkö oppilaat malleja ja mahdollisuuksia toistaa toimivia ohjeita? Osaavatko oppilaat arvioida parhaan reitin? Toimiko ohjelmointi? Tapahtuiko virheiden etsintä loogisesti vai sattumanvaraisesti?

Lukemisen ja äänteiden harjoittelua ohjelmoinnin avulla, 1.lk

Hanna Hämäläinen, CC BY-SA 4.0

30.1.2021

Aihepiirin valinta ja rajaus

Tuntisuunnitelmassani hyödynnetään ohjelmointia äidinkielen asioissa ja tunnit on suunnattu ensimmäisen luokan oppilaille. Ohjelmoinnillisesti harjoitellaan käskyjen antamista ja äidinkielen taitoja harjoitellaan kirjain-äänne-, tavu- tai sanatasolla riippuen oppilaan lukutaidosta. Lukemisen ja äänteiden opettelu vie suuren osan ajasta ja tämän tehtävän olisi tarkoitus tuoda perustunteihin monipuolisuutta lisätä motivaatiota ohjelmoinnin myötä.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Kuten yllä jo totesin, ohjelmoinnillisesti harjoitellaan käskyjen antamista ja äidinkielen taitoja harjoitellaan kirjain-äänne-, tavu- tai sanatasolla riippuen oppilaan tasosta. Ensi harjoitellaan pelkkä ohjelmointiosuus ja sen jälkeen lisätään äidinkielen asiat mukaan.

Oppilaita arvioidaan ensimmäisellä tunnilla käskyjen antamisessa ja toisella tunnilla äidinkielen harjoiteltavan asian hahmottamisessa. Oppilas itse näkee tehtäviä tehdessään onnistumisensa ja mahdolliset virheelliset kohdat, joita voi korjailla oikeiksi.

Työskentelyvälineet ja opetusmenetelmät

Tämän suunnitelman voi muuntaa helposti muihinkin oppiaineisiin ja ympäristöihin toimivaksi. Tässä suunnitelmassa ei käytetä mitään tietoteknisiä välineitä, mutta samankaltaisia ohjeita löytyy BeeBoteille mm. Innokkaan materiaalista (https://www.innokas.fi/wp-content/uploads/2018/02/La%CC%88hto%CC%88laukauskoodaukseen.pdf).

Aiempaa tietoa ohjelmoinnista ei tarvitse olla, mutta kirjaimia pitäisi olla opeteltu. Oppilasryhmästä riippuen ensimmäiseen tuntiin menee noin yksi oppitunti ja seuraavaan voi ohjeineen kulua hieman pidempäänkin tai sen voi jakaa kahteen eri osaan. Jälkimmäinen tunti toimisi parhaiten jakotuntina, jolloin opettaja pystyisi seuraamaan ja ohjaamaan oppilaiden toimintaa paremmin.

Tunti 1 Käskyjen antaminen ja niiden mukaan toimiminen

Tavoite: Harjoitella käskykoodin tekemistä ja ohjeen mukaan toimimista.

Arviointi: Arvioidaan oppilaiden yrittämistä, osallistumista ja käskyjen tekemistä. Arviointi on kannustavaa ja erehdyksestä onnistumiseen painottavaa.

Välineet: Hammastahna ja -harja (tai muuta koodiin tarvittavaa rekvisiittaa)

Toteutus:

1. Ohjaa opettajaa
    a. Oppilaat neuvovat opettajaa hampaiden pesussa (tai jossain vähemmän sotkua aiheuttavassa). Oppilaat antavat siis opettajalle ohjeita ja opettaja toimii niiden mukaisesti.

    b. Keskustellaan ohjeiden antamisen tarkkuudesta (mitä eroa on käskyillä ota hammasharja käteen ja siirrä kättä eteenpäin, avaa kämmen, laita kättä alaspäin kunnes osut hammasharjaa jne.) ja niiden järjestyksestä. Mikä edellisessä kohdassa meni kenties hassusti? Jäikö jotain tekemättä? Voisiko hampaiden pesun hoitaa toisessa järjestyksessä yhtä pätevästi? Tapoja on varmasti hampaiden pesussa oppilailla useita.
2. Ohjaa paria
    a. Sovitaan sopiva tarkkuus ohjeille.
    b. Oppilaat jaetaan pareihin ja he saavat ohjata toisiaan esim. nostamaan kirjan lattialta, kulkemaan luokan ovelle. Laittamaan kynän penaaliin jne. Vaihdetaan vuoroja jokaisen tilanteen jälkeen.
    c. Pohditaan, miten onnistuttiin.
    d. Mietitään, milloin muulloin käskyjä voisi tarvita. Esim. neuvot kaverin tulemaan teille tai reitin kauppaan. Kerrotaan tällainen ohje kaverille.

Tunti 2 Ohjaaminen ruudukossa

Tavoitteet: Äänteiden, tavujen tai sanojen hahmottaminen ja löytäminen, käskyjen tekeminen, käsitteiden oikea ja vasen kertaaminen.

Arviointi: Arvioidaan äänteiden, tavujen tai sanojen hahmottamista ja löytymistä. Koodi on osa harjoitusta, mutta järkevin reitti ja ohjeistus ei ole tässä tehtävässä pääasia vaan äidinkielen teemat.

Välineet: Parille valmiiksi tehdyt ruudukot (esim. 10 ruutua x 10 ruutua), joku pelimerkki ja esine (esim. kynä, kumi), lappuja, joissa harjoiteltuja kirjaimia/tavuja/ sanoja. Ruudukot voivat olla käveltäviä eli suurikokoisia tai sitten voidaan toimia pöytätasossa pienemmällä ruudukolla.

Toteutus:

1. Ohjaaminen ruudukossa: koodaa itsesi kaverin luo
    a. Opettaja jakaa oppilaat suunnilleen samantasoisiin pareihin lukutaidon perusteella.
    b. Parit asettuvat tai asettavat merkkinsä eripuolille ruudukkoa ja vuorotellen tekevät käskysarjoja, jolla pääsevät kaverin luo.
2. Ohjaaminen ruudukossa
    a. Oppilaat sijoittavat kirjaimia/tavuja/sanoja ruutuihin ja toinen asettuu (tai asettaa pelimerkin) johonkin muuhun ruutuun. Ruudukossa oleva/pelimerkin laittanut oppilas sanoo äänteen/tavun/sanan ja pari yrittää keksiä käskysarjan, jolla kyseisen toiminnon saa suoritettua. Ohjeet muotoa liiku yksi x ruutua eteenpäin/oikealla/taakse/vasemmalle.
        i. Eriyttäminen lukutaidon mukaan: kirjain-äänne, tavut tai sanat
        ii. Eriyttäminen ohjelmoinnin mukaan: pienempi tai suurempi alusta, kirjainjonojen etsiminen ilman lukutaitoa

iii. Eriyttäminen molemmissa: isompi matto, sanojen etsiminen kirjaimet oikeassa järjestyksessä

3. Kootaan yhteen onnistumisia.

Kymmenylityksen harjoittelua ohjelmoinnin avulla, 1.lk

Kaija Karlsson, CC BY-SA 4.0

7.4.2021

I tuntisuunnitelma

1. Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Sovellan ohjelmointia matematiikan tuntiin 1.luokalla. Haluan opettaa kymmenylityksen ja pikkulaskun eron ymmärtämistä.

1. Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Havaitaan, ovatko pikkulaskut automatisoituneet. Erityisesti oppilaiden pitää osata kymppiparit hyvin, jotta kympiksi täydentäminen onnistuu. Kymppiparit pitää olla automatisoituneet.

2. Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

Kymmenylityksessä käytetään laskun kirjoittamisessa ns. Kanalatarinan kirjanpitoa, jossa kymppi täydennetään ensin ja sitten lisätään loppu luku.

Kymmenylityksen osaamisen harjoittelua lisää. Tarkistetaan pikkulaskujen osaaminen. Harjoittelussa toiminnallisuus lisää oppimista kaikilla oppilailla, varsinkin kinesteettisillä.

3. Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

Ohjelmoinnissa tavoitteena on oppia vuokaaviota toiminnallisen tehtävän avulla. Vaihtoehdot ovat kyllä ja ei. Siitä edetään ohjeen mukaan. Matematiikassa tavoitteena on harjoitella lisää kymmenylitystä, ja tarkistaa pikkulaskujen ja kymppiparien osaamisen taso: Miten hyvin automatisoituminen on tapahtunut?

4. Miten opettaja tai joku muu arvioi opiskelijan osaamista?

Opettaja havannoi oppilaiden osaamista. Hän katsoo, ketkä laskevat vielä sormilla ja kenellä laskeminen on hidasta, vaikka sormia ei enää käytetäkään. Oppilas voi luetella lukujuja eteenpäin, mikä on vielä poisopittava laskemistapa. Oppilaspari voin havannoida toista oppilasta peukuttamalla tosi hyvin/hyvin/huonosti.

5. Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Arvioidaan erikseen.

6. Miten oppilas arvioi omaa osaamistaan?

Itsearvioinnilla peukuttamalla tosi hyvin/hyvin/huonosti.

Työskentelyvälineet ja opetusmenetelmät

1. Mitä välineitä oppikokonaisuudessa käytetään?

Tarvitaan muotopaloja, teippiä/tikkuja, laskulappuja ja kanalatarinan kirjanpitomoniste

Laskuja laskupurkissa. Laskut ovat kymppipari-, hajotelmalaskuja luvuilla 0 – 9 ja kymmenylityslaskuja yhteenlaskuissa.

2. Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Olemme kanalassa tarkistamassa kanojen munien määriä ja tekemässä niistä kirjanpitoa. Edetään kanalan hoitajan ohjeiden mukaan.

3. Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Kymmenylityksessä oppilaat ovat jo tutustuneet kanatarinaan, jossa on käytetty munakennoja ja helmiä. Kanalalle oppilaat ovat saaneet keksiä nimen. Sitä muistutellaan. Kerrataan vielä aamu- ja iltamunat ja minkä värisiä ne olivatkaan.

4. Miten opetuksessa rohkaistaan yhteistyötä?

Oppilaita rohkaistaan tekemään yhteistyötä niin että toinen tekee ensin ja toinen tarkkailee, meneekö munien laskeminen laskulapuista oikein. Sitten vaihdetaan osia.

5. Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Elämässä on monta kertaa tilanteita, jossa pitää päättää kyllä vai ei ja mitä siitä seuraa. Voidaan keksiä esimerkkejä esim. Menenkö hyppimään lätäkköön ilman kuravaatteita välitunnilla? Kyllä vai ei? Mitä siitä seuraa.

6. Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelu.

Arkipäivän tilanteita voidaan miettiä yhdessä esim. tunteiden hallinta ja kaveruudentaidoissa sekä riidan selvittelyssä.

Mitä teet oppijoiden kanssa?

Lattialla on erivärisiä muotopaloja, joiden päällä lapset seisovat pareittain.

He ottavat purkista taitetun laskulapun, jossa kerrotaan, että kanat ovat munineet aamumunia. He miettivät, onko l**asku?**

oppilas

Pikku lasku	**kymmenylitys**

Oppilas laskee laskun.

pikku lasku, jonka tulos on pienempi tai yhtä suuri kuin kymmenen vai **kymmenylitys.** He valitsevat lattialla olevasta reitistä muotopalan sen mukaan.

Sen jälkeen he laskevat laskun. Sitten he kirjoittavat paperille laskun, jossa on **kanalan kirjanpito kymmenylityslaskuissa**. Siinä näkyy ensin kymppipari ja jäljelle jäänyt luku. Sen jälkeen he laskevat saadut luvut yhteen.

Harjoitusta toistetaan.

II tuntisuunnitelma

Aihepiirin valinta ja rajaus

1. Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Sovellan ohjelmointia suomenkieleen. Haluan oppilaan oppia huomaan pitkät ja lyhyet sanat.

2. Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Pitkät sanat ovat usein yhdyssanoja.

3. Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

Yhdyssana on pitkä, koska kaksi sanaa kirjoitetaan yhteen ja siitä tulee uusi sana.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

1. Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

Ohjelmoinnissa opetellaan lajittelualgoritmilla lajittelemaan sanat annetun ohjeen mukaan. Suomenkielessä opitaan lukemaan ohjeita ja toimimaan niiden mukaan.

2. Miten opettaja tai joku muu arvioi opiskelijan osaamista?

Opettaja havannoi osaamista.

3. Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Osaamista arvioidaan erikseen.

Työskentelyvälineet ja opetusmenetelmät

7. Mitä välineitä oppikokonaisuudessa käytetään?

Ohjeet ovat väripapereilla. Lisäksi tarvitaan sakset, numeropurkkeja, tehtäviä, aapisen teksti ja lista sanoista, joiden perässä on numerot.

8. Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Olemme tietokoneita ja lajittelemme sanoja. Oikea tietokone tekee samaa eli lajittelee, kun haemme vaikkapa kevätkukka sanalla kuvia. Nyt me lajitellaan sanoja kuin tietokoneet.

9. Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Lajittelua on tehty jo ekaluokan ensimmäisillä viikoilla, kun on lajiteltu loogisia paloja eri ryhmiin. Se on tuttua.

10. Miten opetuksessa rohkaistaan yhteistyötä?

Molemmat tarkistavat lajittelun ennen kuin edetään seuraavaan tehtävään. Parilla on yhteisvastuu tekemisestä.

11. Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Loogisten palojen lajittelu on tuttu tehtävä 1.luokan ensimmäisiltä viikoilta matematiikassa.

12. Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelu.

Arkipäivän tilanteita voidaan miettiä yhdessä esim. tunteiden hallinta ja kaveruudentaidoissa sekä riidan selvittelyssä.

Mitä teen oppijoiden kanssa?

Olen valinnut tekstin aapisesta, joka ei ole tuttu oppilaille. Oppilaat toimivat pareittain.

Olemme lajittelukoneita. Oikea tietokone tekee paljon nopeammin kuin ihminen. Nyt harjoitellaan lajittelua.

1. **Näytän aapisen tekstin ja sanon:** - Te leikkaatte monisteesta ensin rivit irti ja sitten leikkaatte sanat irti. Tulee lyhyitä ja pitkiä sanoja.

2. **Punainen ohje:**

   La-jit-te-le sa-nat kah-teen ka-saan:

Alle 3 tavua	3 tai yli 3 tavua

Lai-ta **alle 3 tavua** pois.

3. **Vihreä ohje:**

   Mi-kä on reh-to-rim-me **e-tu-ni-mi**? _____

   Mil-lä kir-jai-mel-la se al-kaa? _____

Sa-na **päät-tyy sa-maan** ki-rjai-meen

sama	Ei sama

Lai-ta ei-ryh-mä pois.

4. **Keltainen ohje:**

   Mi-kä on kou-lu-vii-kon **kes-kim-mäi-nen** päi-vä? _____

   Mil-lä kir-jai-mel-la se al-kaa? _____

   Sa-nas-sa on **sa-ma al-ku-kirj-ain**

sama	Ei sama

Laita ei-ryhmä pois.

5. **Vihreä ohje**

Mi-kä on ryh-män **pi-sin** sa-na? _____

On- ko se yh-dys-sa-na ?

On	Ei

6. **Oranssi ohje:**

Kat-so lis-tas-ta **sa-ma sa-na** ja **nu-me-ro**. O-pet-te-le nu-me-ro.

7. **Sininen ohje:**

Me-ne käy-tä-väl-le. Va-lit-se **purk-ki**, jos-sa on **sa-ma nu-me-ro**.

Ota lappu.Tee teh-tä-vä.

- Tehtävälapuissa lukee

Kaksi sanaa. Näyttele ne niin tulee yhdyssana.

kukka ja ruukku

seinä ja kello

silmä ja lasit

hammas ja harja

korva ja koru

lippa ja lakki

uima ja housut

vesi ja sade

selkä ja reppu

käsi ja voide

Vilske Aapisen teksti

Seit-se-män pe-nin-kul-man saa-pas

Ny-sä Nört-ti kiin-nit-ti et-sin-tä-kuu-lu-tuk-si-a. Hän huo-ma-si mars-si-neen-sa ken-ki-en-sä kär-jet puh-ki. Ka-dun-kul-mas-sa o-li pie-ni ken-kä-kaup-pa ni-mel-tä Seit-se-män pe-nin-kul-man saa-pas. Sin-ne siis. Ny-sä häm-mästyi. Kau-pan myy-jä muis-tut-ti kis-saa. Kis-sal-la o-li pit-kä-var-ti-set saap-paat, pit-kät viik-set, te-rä-vät kyn-net ja o-ve-lat sil-mät. Ei-vät-hän kis-sat myy ken-ki-ä, Ny-sä miet-ti.

Kuin-ka voin pal-vel-la, mjauuu?

Tar-vit-sen uu-det mars-si-ken-gät, kii-tos.

Jaa-ha. Tul-kaa pe-räs-sä-ni e-pä-mu-ka-vi-en jal-ki-nei-den o-sas-tol-le!

III tuntisuunnitelma

Aihepiirin valinta ja rajaus

1. Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Haluan opettaa oppilaille perustietoa salaamisesta. Yhdistän sen suomen kieleen.

2. Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Puhutaan, että tiedon salaaminen on joskus tärkeää, esimerkiksi puhelinnumero ja aikuisilla pankkitiedot. Tietoja voidaan käyttää väärin.

3. Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

Harjoitellaan leikkimieleisesti lukemaan salatekstiä ja laatimaan salakirjoitusta itse.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

1. Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

Tarkka tiedon yhdistäminen etsimällä vastaava merkki ja kirjain. Suomenkielen tavoitteena on tuottaa sanoja ja virkkeitä, joita salataan salakirjoituksella. Harjoitus on tarkkaavaisuuden opettelua.

2. Miten opettaja tai joku muu arvioi opiskelijan osaamista?

Havannoimalla työskentelyä.

3. Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Osaaminen arvioidaan yhdessä.

4. Miten oppilas arvioi omaa osaamistaan?

Peukuttamalla tosi hyvä/hyvä/huono

Työskentelyvälineet ja opetusmenetelmät

13. Mitä välineitä oppikokonaisuudessa käytetään?

- Monisteet, joissa salakirjoituksen kuvat ja vastaavat merkit.

Etsi vastaava merkki.

€ £ ∞ ± © Δ ☺ µ ® Ω α Σ ¥ β ₹ ♥ ≥ × ÷ a e h
i j k l m n o p r s t u v y ä ö

14. Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Nyt ollaan hiljaa. Kukaan ei saa kuulla tai nähdä, että olemme koulussa. Hetki hiljaa. Tällaista on salaaminen ja salassaolo. Ei paljasteta kenellekään asioita. Joskus se on tärkeää. Keskustellaan, mitä asioita pitää salata. Esim. Oppilailla pitää olla salasana Googleen kirjautuessa tai heillä pitää olla pin-koodi avatessaan puhelimen lukitusta.

15. Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Salaaminen on mystistä. Alkutarina, johon keksitään salanimi päähenkilölle. Päähenkilö halutaan salata. Tarina omasta aapisesta, jota parit lukevat toisilleen.

16. Miten opetuksessa rohkaistaan yhteistyötä?

Oppilaita rohkaistaan tekemään yhteistyötä niin että toinen tekee ensin ja toinen tarkkailee, meneekö oikein. Vaihdetaan osia.

17. Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Lahjan valmistaminen koulussa, josta ei saa puhua. Salataan tieto.

18. Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelu.

Salasanan tekemisestä ei saa kertoa kenellekään. Tieto pitää olla vain itsellä. Tieto salataan. Voidaan opettaa oppilaille tarpeeksi turvallisen salasanan keksimiseen, joka kuitenkin muistaisi. Lorun alkutavut ja erikoismerkit ja kirjaimet –tyyppinen harjoitus.

Mitä teen oppijoiden kanssa?

1. Keskustelu salauksesta. Johdantomateriaali kohdan 14 mukaan.
2. Opettaja mallittaa salasanojen lukemisen.
3. Oppilaat ottavat selvää 5 – 10 salasananasta tai –lauseesta, jotka opettaja on kirjoittanut etukäteen kohdan 13 merkeillä.
4. Oppijat keksivät parilleen salasanoja tai –lauseita.
5. Itse- ja kaveriarviointi peukuttamalla työskentely innokkuudesta sekä sanojen ja lauseiden keksimisestä.

Ohjelmointia LEGO-roboteilla, 1.lk

Daria Korkka, CC BY-SA 4.0

18.4.2021

Aihepiirin valinta ja rajaus

Toimin tällä hetkellä ensimmäisen luokan opettajana, joten päätin suunnitella kokonaisuuden alkuopetukseen sopivaksi. Osallistuimme luokkani kanssa STEAM junior -haasteeseen, joka kuuluu Turun kaupungin STEAM-hankkeeseen (https://www.turku.fi/steam-turku/steam-junior-haaste). STEAM-hankkeen tarkoituksena on lisätä tekniikan alan koulutustarjontaa ja tutkimusta Turun kaupungissa. Tämän vuoden junior-haasteen pienimpien oppilaiden tehtävänä on suunnitella ja rakentaa legoista robotti ja sille esterata. Robottia tulee ohjata kulkemaan esteradalla ja kuvata tästä kilpailuvideo. Ajattelin suunnitella kokonaisuuden tämän aiheen ympärille, jotta pääsen hyödyntämään suunnitelmaa luokassani mahdollisimman pian. Suunnitelman voi kuitenkin toteuttaa ilman kilpailuakin, jos käytössä on legoja tai muita materiaaleja, joista voidaan rakentaa robotti ja esterata.

Rakennamme robottia ja esterataa käsityön, kuvataiteen ja ympäristöopin tunneilla, mutta itse ohjelmointi liittyy tietenkin matematiikkaan. Ohjelmointia tulisi Perusopetuksen opetussuunnitelman mukaan (Opetushallitus 2016) harjoitella jo alkuopetuksen aikana ja tässä projektissa siihen päästään tutustumaan monipuolisesti ja luomaan itse kiinnostavaa ympäristöä. Projekti on hyvin monialainen, kun lisäksi harjoitellaan ryhmässä työskentelyä, videon kuvaamista sekä videolla esiintymistä. Käytännössä projektiin saa yhdistettyä ominaisuuksia lähes mistä tahansa oppiaineesta.

Projektin tavoitteena on harjoitella suuntia (vasen ja oikea), yksinkertaista ohjelmointia ilman koneita ja laitteita sekä ohjeiden seuraamista tarkoituksenmukaisesti. Lisäksi projektissa harjoitellaan erityisesti ryhmässä työskentelyä; toisen kuuntelua ja omien mielipiteiden ilmaisua.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppilaiden tavoitteena on kerrata ja harjoitella ohjelmointia, sekä harjoitella ryhmässä toimimista. Opettaja arvioi oppilaiden osaamista seuraamalla työskentelyä. Tässä projektissa keskityn arvioimaan ohjelmointiosaamista, sillä olemme harjoitelleet sitä jo aiemmin. Nyt on hyvä aika seurata ja arvioida, miten hyvin oppilaat hallitsevat ohjelmoinnin perusteet.

Jokaisen työskentelykerran jälkeen reflektoidaan omaa oppimista ja työskentelyä. Työskentelyn jälkeen opettaja esittää väittämiä, jotka jokainen arvioi peukulla. Väittämiä voi esittää lisääkin ja eri työskentelykerroilla voidaan arvioida hieman eri asioita riippuen siitä, mitä on tehty. Alla kaksi väittämää, jotka arvioidaan joka kerralla.

- Tein parhaani ja osallistuin yhteiseen tekemiseen niin hyvin kuin osasin.

- Kuuntelin muita ja osasin ottaa ryhmäläiseni huomioon.

Työskentelyvälineet ja opetusmenetelmät

Tässä projektissa jokaisella pienryhmällä on käytössään paketti legoja. Muuten robotin ja esteradan rakentamiseen saa käyttää mitä vain koulusta löytyviä materiaaleja, kuten kartonkia, massapalloja, helmiä ja muita askartelutarvikkeita. Oppilaiden käytössä on myös puutavaraa, kuten erilaisia palikoita.

Työskentely aloitetaan kertaamalla ohjelmoinnin perusteet. Muistellaan, millaisia käskyjä annetaan ja miten niiden mukaan toimitaan; nuoli eteenpäin tarkoittaa eteenpäin liikkumista, käännös tarkoittaa pelkästään kääntymistä, minkä jälkeen täytyy taas jatkaa eteenpäin ja nuoli taaksepäin tarkoittaa taaksepäin liikkumista. Nämä perustoiminnot ovat ryhmälle jo ennestään tuttuja, mutta ne kerrataan ennen työskentelyn aloittamista. Kertauksen jälkeen oppilaille kerrotaan, että seuraavaksi tehtävänä on suunnitella ja rakentaa robotti ja sille esterata. Heitä kannustetaan pohtimaan, millainen robotti on ja millainen esterata juuri tälle robotille kannattaisi rakentaa.

Oppilaita motivoidaan kertomalla ja näyttämällä esimerkkejä erilaisista roboteista. Oppilaita motivoi varmasti myös kilpailu, johon osallistutaan, sillä sen palkintona on yhteinen tekeminen koko luokalle. Yhteistyöhön kannustetaan alusta asti ja korostetaan, että ilman toimivaa yhteistyötä ei voi tehtävässä onnistua. Yhteistyön onnistumista arvioidaan jokaisen työskentelykerran jälkeen ja kannustetaan toimintaan, joka edesauttaa yhteistyön onnistumista. Opettaja kertoo konkreettisia esimerkkejä, joita oppilaat voivat käyttää ryhmässä työskennellessään, kuten lauseita, joita voi ryhmäläisille työskentelyn aikana sanoa.

Jos ryhmässä on oppilas, jolla on enemmän ohjelmointiosaamista, ryhmä voi käyttää osaamista vapaasti hyödyksi. Tehtävä on kovin avoin, joten monenlaisesta osaamisesta on hyötyä, kunhan osaamisen käyttöön kannustetaan. Koska erilaisia materiaaleja saa käyttää melko vapaasti, pääsee monenlainen osaaminen käyttöön.

Tämän oppikokonaisuuden aikana sekä sen jälkeen ohjelmoimista jatketaan luokassa erityisesti matematiikan tunneilla. Tutustutaan esimerkiksi erilaisiin ohjelmointiohjelmiin, joita löytyy koulun iPadeista, kuten Scratch-junioriin.

Oppikokonaisuuden suunnitelma

Ensimmäisen tunnin aluksi kerrataan ohjelmoinnin peruskäsitteet ja palautetaan mieleen, miten erilaisia käskyjä annetaan ja miten niiden mukaan toimitaan. Opettaja näyttää kilpailun mukana tulleen ohjeistuksen ja käskyjen mukaan toimimista

harjoitellaan toiminnallisesti niin, että opettaja näyttää kortteja ja oppilaiden on liikuttava käskyjen mukaan. Korteissa ohjeistetaan liikkumaan mm. eteenpäin, taaksepäin ja kääntymään.

Ohjelmoinnin perusteiden kertaamisen jälkeen annetaan ohjeistus työskentelyyn. Oppilaille kerrotaan, että tehtävänä on suunnitella ja rakentaa legoista sekä mahdollisesti muista tarvikkeista robotti ja sille esterata. Esteradan on oltava sellainen, jossa on esimerkiksi ruutuja tai muita selkeitä pysäkkejä, jotta robotti pystyy noudattamaan sille annettuja käskyjä ja liikkua esimerkiksi kaksi ruutua eteenpäin. Lopuksi robotin on tarkoitus selvitä esteradasta ohjaamalla se toimimaan tarkoituksenmukaisesti.

Oppilaille annetaan paperi suunnittelua varten, legot ja kerrotaan, mitä muita mahdollisia materiaaleja on saatavilla. Pienryhmissä oppilaat saavat tutkia materiaaleja ja suunnitella, millaisen robotin rakentavat. Työskentelyä jatketaan noin kolmen viikon verran käsityön, kuvataiteen, ympäristöopin sekä mahdollisesti matematiikan tunneilla. Lopuksi robotista ja sen ohjaamisesta esteradalla kuvataan kilpailuvideo.

Oppikokonaisuus voidaan toteuttaa tällaisena ilman kilpailua. Itse toteutan suunnitelman kilpailun osana, joka itsessään motivoi oppilaita osallistumaan ja tekemään parhaansa. Uskoisin kuitenkin, että robotin ja esteradan suunnittelu ja rakentaminen ovat monen oppilaan mieleen, vaikka mitään kilpailua ei olisikaan.

Äidinkieltä ja ohjelmointia, 1.lk

Nimetön,

22.2.2021

- Lähestyn ohjelmointia äidinkielessä puhumisen ja kuuntelemisen kautta, sillä ne ovat tärkeitä taitoja koululaisille
- Aihepiiriini liittyy ilmiönä koululaisen arkeen ja tunnilla toimimisen haasteisiin sekä itsensä ilmaiseimseen
- Oppimistavoitteina näissä ilmiössä on se, että oppilaan tulisi oppia kuuntelemaan ja toimimaan ohjeiden mukaan sekä ilmaisemaan itseään sanallisesti
- Ohjelmoinnin näkökulmasta tavoitteena on oppia ohjeiden noudattamista, kuten myös äidinkielen näkökulmasta. Äidinkieleen liittyen tavoitteena on myös oppia ilmaisemaan itseään
- Opettaja arvioi osaamista oppilaan tehtävien tekemisen pohjalta. Koska osaamistavoitteet menevät päällekkäin, arvioidaan osaamista kokonaisuutena
- Oppilas arvioi omaa osaamistaan peilaten tehtävissä onnistumiseen ja niistä tulevaan suoraan palautteeseen

- Oppikokonaisuuteen tarvitaan koulun jumppasalia, huiveja ja narua, penkejä jne
- Kokonaisuuteen lähdetään sillä, että oppilaiden kanssa keskustelaan luokassa millaisia erilaisia ohjeita he tietävät ja mitä taitoja he tarvitsevat erilaisten ohjeiden noudattamiseen. Oppilaita johdatellaan siihen, että ohjeita pitää kuunnella ja noudattaa tarkasti. Sitten keskustellaan siitä, miten itse annat ohjeita ja sitä harjoitellaan luokassa pareittaan niin, että pyydetään toista hakemaan jokin tavara ilman, että tavaran nimeä saa sanoa.
- Oppilaat motivoidaan tehtävään leikin varjolla ja oman hoksaamisen kautta
- Opintokokonaisuuden tehtävät tehdään pareittain ja niissä vaaditaan paljon yhteistyötä.
- Oppilas voi tuoda aiempaa osaamistaan esille niin, että hän osaa jo toimia tehtävissä pyydettyjen ohjeiden mukaisesti ja tuntee leikkien säännöt
- Oppikokonaisuuden jälkeen ohjelmointiin voidaan lähteä tutustumaan niin, että käytetään ohjeiden noudattamisen ja antamisen taitoja yksinkertaisissa tietokoneilla tehtävissä ohjelmointitehtävissä

1. oppitunti 45 min

· Käydään luokassa edellä esitetty alkukeskustelu ja ensimmäinen harjoitus

· Pelataan kapteeni käskee -peliä, opitaan ohjeiden kuuntelemista

· Mietitään yhdessä, miksi ohjeiden kuuntelu on välillä vaikeaa

2. oppitunti 45 min

· Siirrytään koulun jumppasaliin parijonossa, jonka liikkumista opettajaa ohjeistaa (hiivitään, laahustetaan, kävellään rauhallisesti yms.)

· Nopeasti kerrataan ohjeiden kuuntelun tärkeimmät asiat

· Pareittain harjoitellaan ohjeiden antamista: toisella on silmät sidottuna ja hänet pitää ohjata kävelemään parin luokse

· Opettaja tekee lattialle köydestä, penkeistä tms labyrintin. Oppilaat ohjaavat pareittain toisen kulkemaan labyrintin läpi silmät sidottuna

· Mietitään yhdessä ohjeiden antamista ja mikä siinä oli helpointa/vaikeinta

Kymppiparien harjoittelu ohjelmoinnin avulla, 1.lk

Taija Caselius, CC BY-SA 4.0

26.11.2021

Tuntisuunnitelmani on suunniteltu matematiikkaan ja ohjelmoinnin avulla harjoitellaan kymppipareja. Valitsin aiheen, koska kymppiparit ovat keskeinen osa ekaluokan matikan opetusta, osalle ne ovat todella vaikeita ja haluaisin itselleni mahdollisimman monipuolisen vinkkipaketin niiden opettamiseen toiminnallisesti. Oppilaani eivät ole aikaisemmin tehneet ohjelmointia, joten lähdemme ohjelmoimaan ilman laitteita. Harjoittelemme käskyn antamista ja käskyjonon tekemistä. Ajatuksena on, että ensimmäinen tunti toimitaan koko ajan pareittain ja toisella tunnilla on enemmän yksilötoimintaa. Oppimistavoitteiksi asetin parin kanssa työskentelyn (omalle porukalleni aika haastavaa) ja kymppiparien opettelun.

Arvioinnista

Tavoitteena on harjoitella parin kanssa työskentelyä, toisen huomioimista, kymppipareja, käskyn antamista robotille. Tavoitteiden arviointi tapahtuu pääasiassa työskentelyn havainnointina, kymppiparien osaaminen myös kirjallisena tuottamisena. Aion keskittyä näillä tunneilla itse- ja vertaisarviointiin ja pyydän jokaisen tehtävän jälkeen oppilaita antamaan palautetta parille tai näyttämään peukulla arvion omasta osaamisestaan.

Työskentelyvälineet ja opetusmenetelmät

Välineiksi tunneilla tarvitaan multilink -palikoita, jonkinlaiset numerokortit 0-10, (valmiiksi mietittyjä laskuja n. 6kappaletta, ryhmästä riippuen), joiden vastaukset matematiikan kirjat/moniste, jossa on kymppiparitehtäviä.

Suunnitelma

Ensimmäinen tunti aloitetaan parityöskentelynä. Kerron oppilaille tunnin tavoitteet (parin kanssa työskentely ja kymppiparien harjoittelu). Jokaisella oppilaalla on kymppipötkö multilink-palikoita siten, että pareilla on eriväriset pötköt. Toinen pareista aloittaa koneena ja toinen on robotti, joka laittaa koneeseen palikoita ja sanoo ulostulevan hajotelman niin, että sanoo ensin itselaittamiensa palikoiden määrän. Kerron oppilaille, että kone on ohjelmoitu siten, että sieltä tulee ulos vain täysiä kymppipötköjä. Robotti siis antaa koneelle pötkön palikoita, kone laittaa kympin täyteen ja robotti on siirtyy toiselle puolelle konetta, ottaa ulos tulevan pötkön vastaan ja sanoo ääneen hajotelman. Tätä tehdään niin kauan, että molemmat ovat saaneet toimia koneina ainakin 5 kertaa. Toiminnan päätteeksi pyydän oppilaita kertomaan kaverille, mikä hänellä meni hyvin.

Seuraavaksi tehdään pareittain hommia siten, että toinen on robotti ja toinen antaa käskyjä. Ensimmäinen käsky on, että robotin on mentävä lattialla olevista numerolapuista sen luokse, jonka luku pitää lisätä käskynantajan lukuun, että saadaan kymppi täyteen. Ensimmäinen käskynantaja sanoo 5 lukua ja osat vaihtuu. Toinen käsky on se, että robotin on mentävä sen numerolapun luokse, jonka luku jää jäljelle, kun luvusta 10 vähennetään käskynantajan sanoma luku. Tehtävän

päätteeksi jokainen näyttää peukulla, miten hyvin yhteistyö parin kanssa omasta mielestä sujui.

Lopputunnin jokainen oppilas on robotti, joka tekee kirjan tehtäviä. Robotti ei kuitenkaan saa siirtyä seuraavaan tehtävään ennen kuin opettaja on tarkistanut edellisen tehtävän ja robotti on käynyt hyppäämässä taululla olevan luvun kymppiparin verran XI-hyppyjä luokan perällä.

Jokerina voisi olla, että välitunnille pääsee, kun robotti on käynyt kuiskaamassa opelle open antaman luvun kymppiparin.

Toinen tunti alkaa sillä, että jokainen oppilas on robotti. Opettaja ohjelmoi robotit hyppäämään niin monta hyppyä, että kymppi tulee täyteen opettajan hyppyjen kanssa eli ope hyppää esim. 3 hyppyä ja oppilaat hyppäävät 7.

Hyppelyn jälkeen opettaja ohjelmoi robotit laskemaan open antamat laskut ja siirtymään esim. luokan perälle, jos vastaus on alle 10, luokan keskelle, jos vastaus on tasan 10 ja luokan eteen jos vastaus on suurempi kuin 10. Tähän tehtävään pystyy hyvin yhdistämään myös päässälaskut. Robotit voi ohjelmoida myös käyttämään palikoita apuna. Tämän tehtävän jälkeen jokainen arvioi, miten hyvin osaa kymppiparit tällä hetkellä, esim. peukulla näyttämällä.

Viimeinen robottitehtävä on, että robotit on ohjelmoitu kirjoittamaan taululla olevista luvuista yhteenlaskuja, joiden vastaus on tasan 10. Tästä voi tehdä myös helpomman version siten, että taululla on valmiita laskuja ja robotit kirjaavat ne laskut, joiden vastaus on 10.

Jokaisessa toiminnallisessa tehtävässä opettajalla on mahdollisuus havainnoida oppilaiden osaamista ja kehittymistä. Jälkimmäisen tunnin lopuksi voisi olla hyvä pohtia, miten robottitehtävät erosivat yleensä matikan tunneilla tehtävistä tehtävistä ja missä muualla he ovat toimineet ikään kuin robotteina. Samalla voisi pohjustaa ajatusta, että robotti on vähän niin kuin tietokone tai kännykkä, jolle annetaan käskyjä ja joka noudattaa niitä. Jonkin robottitehtävän yhteydessä voidaan myös kokeilla, mitä tapahtuu, jos ohjelmoija antaa sellaisen käskyn, ettei robotti pysty noudattamaan sitä esim. mene luvun 15 kymppiparin luo.

Tupiin jako ja Kalkaroksen liemitunti -pakohuonepeli, 1.-2. lk

Mirva Slunga, CC BY-SA 4.0

19.11.2021

LUMATIKKA 3:

Mirva Slunga syksy 2021

Algoritmisen ajattelun kehittäminen

Tuntisuunnitelma-työpaja: Tupiin jako ja Kalkaroksen liemitunti -pakohuonepeli

Aihepiiristä

Luokassa on ollut menossa Harry Potter –jakso. Olemme tehneet teemaan liittyen monialaisesti erilaisia tehtäviä matematiikan, äidinkielen, käsityön ja kuvataiteen tunneilla. Tämä tuntisuunnitelmani on jakson päätös. Suunnitelmani sisältää noin kahden oppitunnin pituisen kokonaisuuden. Ensimmäisessä osassa oppilaat jakautuvat 3-4 hengen ryhmiin ominaisuuksiin liittyvän tehtävän avulla. Toisessa osassa luokkaan on koottu pakohuonepeli, jossa tarvitaan aiemmin opittuja ohjelmointiin liittyviä taitoja, päättelykykyä, algoritmista ajattelua ja ongelmanratkaisutaitoja. Pakohuonepeli on tarkoitus ratkaista yhden oppitunnin aikana. Opettaja voi tarvittaessa taitavasti ohjailla oppilaita, jotta jokainen selviytyy ja onnistuu tehtävässä (alkuopetuksessa on ehkä kiva että kaikki läpäisevät tehtävän).

Pakohuonepelin valinta tuntisuunnitelman pohjaksi johtuu omasta innostuksestani. Olen kerran aiemmin toteuttanut yhden pakohuoneen Kalevala-teemalla. Halusin kokeilla ja harjoitella pakohuoneen muodostamista nyt uudelleen. LUMATIKKA 3-kurssin sisällöt tuntuivat sopivan pakohuonepeliin hyvin. Sain muodostettua kokonaisuuden, jossa oppilailla oli mahdollista käyttää hyväksi aiemmin opittuja ja kokeiltuja asioita ja sen lisäksi soveltaa niitä uudenlaisissa tehtävissä. Suunnittelemani tuntikokonaisuus toimi myös hyvänä oppilaiden havainnoinnin välineenä itselleni.

Kohderyhmä

Alkuopetuksen oppilaat, meillä 1.- ja 2.-luokkalaisten yhdysluokka.

Mielestäni tehtävä on sovellettavissa myös isommille oppilaille. Sen perusrunkoa voi käyttää hyödyksi ja tarvittaessa vaikeuttaa ratkaisuun liittyviä tehtäviä.

Oppimistavoitteet (matematiikan osalta) opetussuunnitelmasta poimien

”Merkitys, arvot ja asenteet
T1 tukea oppilaan innostusta ja kiinnostusta matematiikkaa kohtaan sekä myönteisen minäkuvan ja itseluottamuksen kehittymistä
Työskentelyn taidot
T2 ohjata oppilasta kehittämään taitoaan tehdä havaintoja matematiikan näkökulmasta sekä tulkita ja hyödyntää niitä eri tilanteissa
T3 kannustaa oppilasta esittämään ratkaisujaan ja päätelmiään konkreettisin välinein, piirroksin, suullisesti ja kirjallisesti myös tieto- ja viestintäteknologiaa hyödyntäen
T4 ohjata oppilasta kehittämään päättely- ja ongelmanratkaisutaitojaan
Käsitteelliset ja tiedonalakohtaiset tavoitteet
T5 ohjata oppilasta ymmärtämään matemaattisia käsitteitä ja merkintätapoja
T12 harjaannuttaa oppilasta laatimaan vaiheittaisia toimintaohjeita ja toimimaan ohjeen mukaan

Matematiikan tavoitteisiin liittyvät keskeiset sisältöalueet vuosiluokilla 1–2
S1 Ajattelun taidot: Oppilaille tarjotaan mahdollisuuksia löytää yhtäläisyyksiä, eroja ja säännönmukaisuuksia. Vertaillaan, luokitellaan ja asetetaan järjestykseen sekä havaitaan syy- ja seuraussuhteita. Harjoitellaan tarkastelemaan matemaattisia tilanteita eri näkökulmista. Tutustuminen ohjelmoinnin alkeisiin alkaa laatimalla vaiheittaisia toimintaohjeita, joita myös testataan.

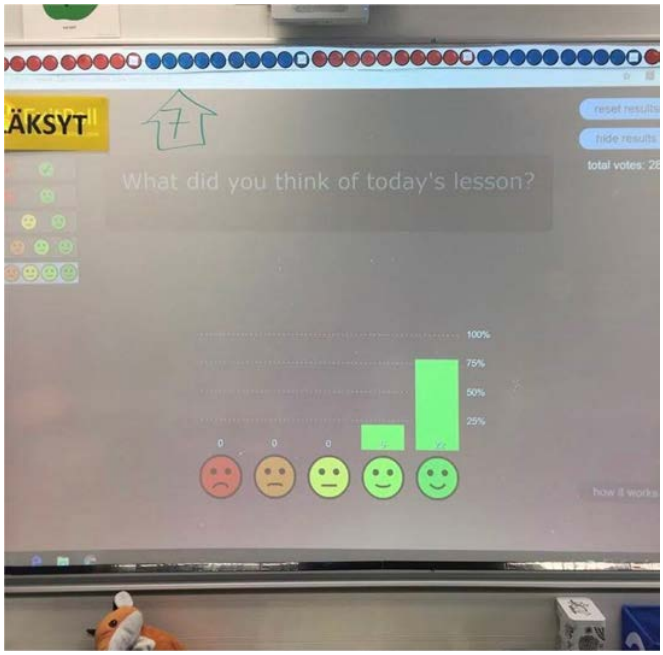
Matematiikan oppimisympäristöihin ja työtapoihin liittyvät tavoitteet vuosiluokilla 1–2
Opetuksen lähtökohtana käytetään oppilaille tuttuja ja kiinnostavia aiheita ja ongelmia. Tavoitteena on luoda oppimisympäristö, jossa matematiikkaa opiskellaan toiminnallisesti ja välineiden avulla. Opetuksessa käytetään vaihtelevia työtapoja. Oppilaat tottuvat työskentelemään sekä itsenäisesti että yhdessä. Pedagogisesti ohjatut leikit ja pelit ovat yksi tärkeä työtapa. Opetuksessa ja opiskelussa käytetään tieto- ja viestintäteknologiaa."

(Opetushallitus 2014, 128–131.)

Lisäksi suunnittelemani kokonaisuuden tavoitteena oli harjoitella ymmärtävää lukemista, ja kiinnittää huomiota virkkeen sisällä oleviin sanoihin, tavuihin ja kirjaimiin.

Arviointi

- Opettajan jatkuva havainnointi
- Tehtävissä onnistuminen heijastaa osaamista. Oppilas saa tehtävistä välitöntä palautetta onnistuessaan ja päästessään jatkamaan eteenpäin. Opettaja tarvittaessa ohjaa ja kannustaa tehtävissä etenemisessä. Myös pari voi auttaa ja neuvoa.
- Kokonaisuuden lopuksi koonti classroomscreenillä ja yhteistä keskustelua: Oppilaat vastaavat hymiöillä Miltä tehtävä tuntui? Miten osasin?



Kuva 1: Classroomscreenin näkymä taululla

OSA 1 (paikalla koko luokka): joukkueisiin eli tupiin jako

Välineet:

- Opettaja tulostaa etukäteen valmiiksi 3-4 samanlaista pöllökorttia/tupa pöllöjen loogisesta kokoelmasta
- oppilaiden päähän tulevat pannat, joihin pöllökortin voi kiinnittää

Pöllöjen looginen kokoelma sisältää kuvia pöllöistä, joilla on tietyt ominaisuudet. Nämä ominaisuudet ovat häkillisyys (on häkki/ei ole häkkiä), kirjeellisyys (on kirje/ei ole kirjettä) ja paras lentoilmansuunta (pohjoinen/itä/etelä/länsi).

Kuva 2: Pöllöjen looginen kokoelma

Tehtävän kuvaus

Oppilaat jakautuvat eri joukkueisiin eli tupiin pöllöjen ominaisuuksiin liittyvällä tehtävällä. Yhteen tupaan tulee 3-4 oppilasta. Opettaja voi oikeasti arpoa tuvat, tai sitten ”arpoa” eli jakaa oppilaat ryhmiin erilaisin pedagogisin perustein.

Looginen kokoelma pöllöistä on oppilaille tuttu. Olemme tarkastelleet aiemmin pöllöjen ominaisuuksia ja tehneet erilaisia logiikkaan liittyviä tehtäviä kokoelman parissa. Jokainen oppilas saa yhden pöllökortin. Korttia ei saa kurkata vaan se kiinnitetään otsaan. Tehtävänä on etsiä luokkakaverit, joilla on juuri samanlainen pöllö kuin itsellä. Oppilas saa kysyä pöllöstään kysymyksiä, ja luokkakaveri vastaa kyllä tai ei.

Kun kaikilta ominaisuuksiltaan samanlainen pöllö löytyy, kaveria otetaan kädestä kiinni. Samanlaisen pöllön omistajat muodostavat yhden tuvan.

Kuva 3: Oman pöllön ominaisuuksien ratkaisemista kaverilta kysyen

Tietyt oppitunnin toisen osan tehtävät on värikoodattu tupien mukaan. Toisten tupien väritehtäviin ei saa koskea.

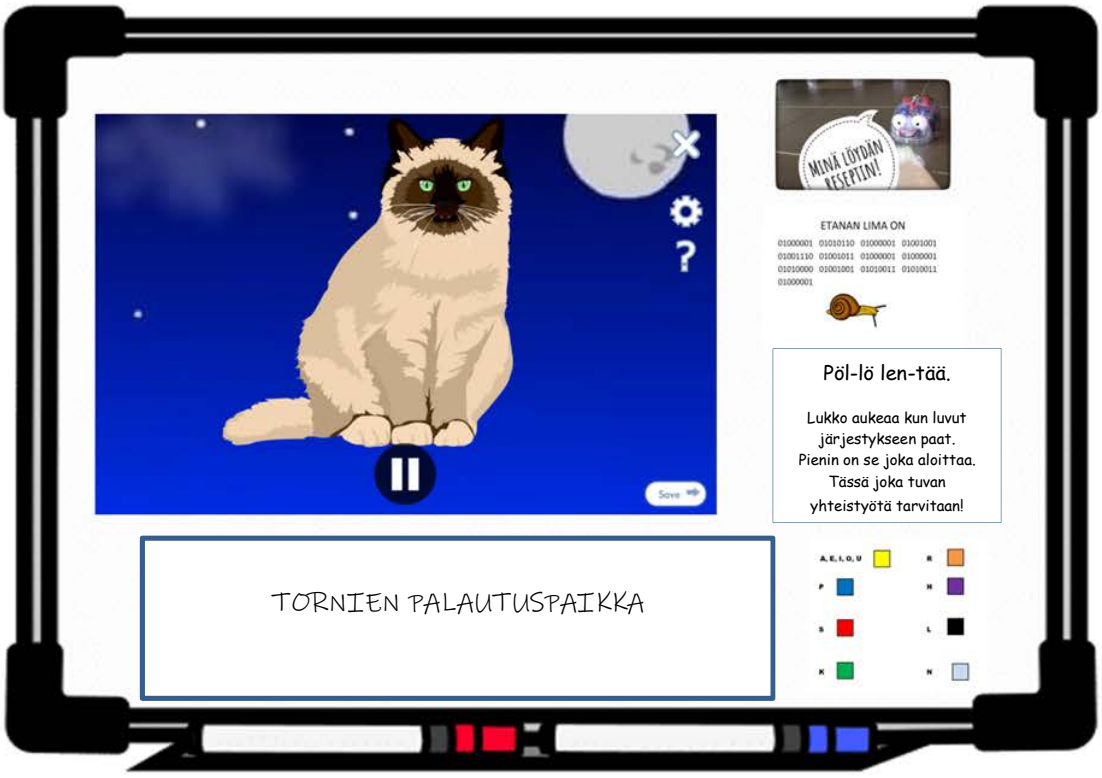
Kuva 4: Tuvat ja niiden värit

Punaiset -Rohkelikko
Keltaiset - Puuskupuh
Siniset - Korpinkynsi
Vihreät – Luihuinen

OSA 2(paikalla puolikas ryhmä): Kalkaroksen liemitunti – alkuopetuksen pakohuonepeli

Etukäteisvalmistelut pakohuonepeliin:

- Opettajalle velhoviitta ja muuta rekvisiittaa
- Classroomscreeniin kysely valmiiksi
- Luokan edessä isompi pöytä, joka on suojattu jätesäkeillä. Pöydällä iso kyltti, jossa lukee ”LIEMENVALMISTUSPAIKKA”. Pöydälle 4 x pilttipurkki (isompi), 4 x teelusikka ja 4 x ruokalusikka, 4 x kulho yksisarvisen kyyneliä eli etikkaa varten
- Taululle esille:
    - kuva Bluebotista ja sillä on puhekupla: ”Minä löydän reseptin!”
    - etanavihje eli kuva etanasta ja binäärikirjoitusta
    - duplorakentamisen avain eli moniste, jossa on kirjaimia ja niitä vastaavia väripalikoita
    - moniste jossa lukee ”Pöl-lö len-tää. Lukko aukeaa kun luvut järjestykseen paat. Pienin on se joka aloittaa. Tässä joka tuvan yhteistyötä tarvitaan!”
    - tussilla rajattu alue, johon rakennetut duplotornit palautetaan ja teksti ”TORNIEN PALAUTUSPAIKKA”



Kuva 5: valmisteltu taulukuva

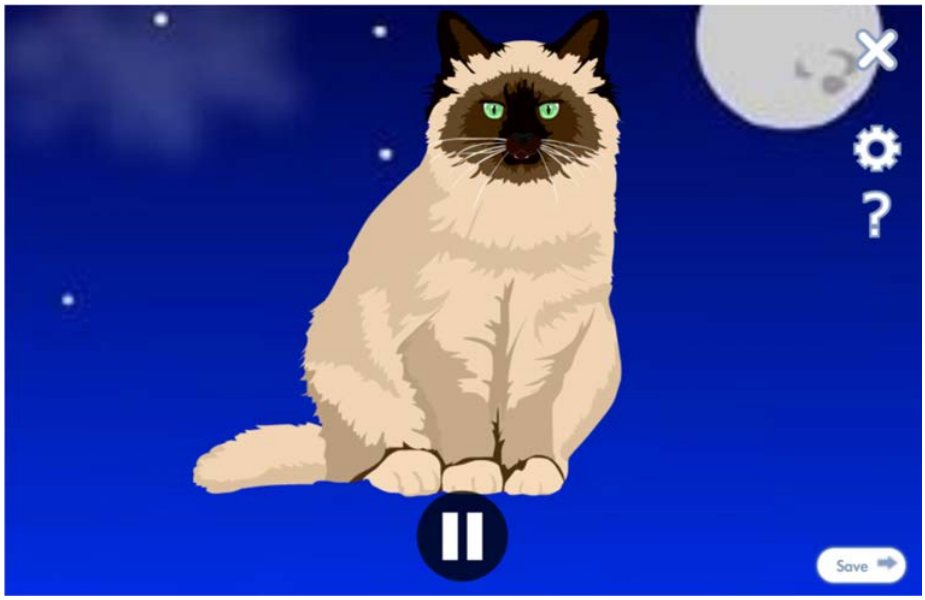
- Opettajan pöydän rajaaminen esimerkiksi paksulla langalla (alue, johon ei saa mennä)
- Opettajan pöydän taakse hyllylle 4 pilttipurkkia, jokaiseen liskonaivojauhoja eli soodaa

- Pöydille tai luokkaan lattialle valmiiksi:
    - 4 Bluebot-robottia
    - 4 Bluebotin lattiaruudukkoa
    - 4 liemiaineslistaa, yksi jokaiseen lattiaruudukkoon yhteen ruutuun valmiiksi paikoilleen
    - taskulamppuja ja purkkien kuvia läpivalaisua varten. Purkeissa liskon kuvia ja tekstiä: ”Liskonaivojauhot ovat open hyllyllä” ja ”Ope auttaa ja hymyilee, kun sille kahvia tarjoilee!”
    - pikseliväritystehtävämonisteita ja kyniä
    - 4 käenmunatehtävää loogisista liemipulloista, jokaiselle tuvalle omansa
- 4 monistettua binääriavainta
- 4 pilttipurkkia etananlimaa eli fairya taulun eteen avainkaappiin. Kaapit lukkoon.
- 4 avainta, jotka sopivat taulun alapuolella olevien kaappien lukkoihin
- Jokaisen etananliman yhteyteen lappu, jossa lukee ”Rakenna, rakenna, tuvan nimi duploista rakenna! Ja torni taululle palauta!”
- duplopalikoita (Varmista että jokaista väriä on riittävästi tehtävää varten)
- 4 elintarvikeväripulloa, jokaiseen kiinni yksi tehtävälappu:

    Rohkelikon punainen paperi: ”Kuinka monta kirjainta on ensimmäisessä sanassa?”
    Korpinkynnen sininen paperi: ”Kuinka monta kirjainta on toisessa sanassa?”
    Puuskupuhin keltainen paperi: ”Kuinka monta tavua?”
    Luihuisen vihreä paperi: ”Kuinka monta sanaa?”
- pyörän lukko, johon tulee neljä numeroa
- pullo etikkaa kaappiin pyörän lukon taakse. Laita pulloon Yksisarvisen kyyneliä –nimilappu.

Tunnin aloitus ja tehtävän motivointi

Oppilaat ovat Kalkaroksen liemitunnilla. Heidän tulee saada tietty liemi valmiiksi ennen kuin tunti päättyy tai he jäävät jälki-istuntoon. Tunti alkaa yhteisellä osuudella. Oppilaat tulevat istumaan edellisellä kerralla jaettujen tupakavereiden kanssa taulun eteen, ja katsovat videotervehdyksen.



Kuva 6: Vokilla tehty velhokissan video

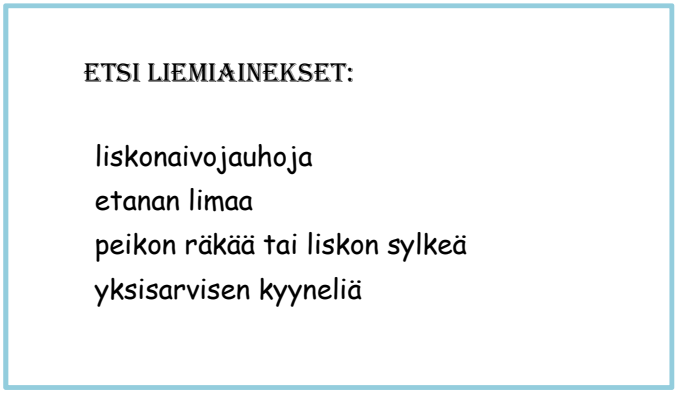
Videon velhokissa juttelee:

”Tervetuloa professori Kalkaroksen liemitunnille! Professori on työmatkalla, ja luokaan on saapunut apulaisopettaja teitä valvomaan. Teidän tehtävänä on valmistaa taikaliemi ohjeen mukaan, tai muuten jäätte jälki-istuntoon! Kaikkeen muuhun saa luokassa koskea, mutta opettajan pöydän taakse ei saa mennä. Tarvitsette huolellisuutta, ongelmanratkaisua ja yhteistyötä! Etsikää ensin resepti! Olkaa nopeita. Aikaa on vain tämä tunti! Onnea matkaan! Aika alkaa ÄN-YY-TEE-NYT!”

Oppilaat tietävät mitä pakohuonepeli tarkoittaa. Aiheesta on käyty aiemmin keskustelua. Opettaja voi vielä käydä ohjeen selkeästi läpi. Tarvittaessa velhokissan video voidaan katsoa uudelleen. Opettajan tehtävänä on pakohuonepelin aikana seurata oppilaiden työskentelyä ja tarvittaessa hienovaraisesti ohjailla heitä eteenpäin. Kelloa kannattaa seurata tarkoin, jotta jokainen lapsi varmasti ehtii tehtävän ratkaista.

Pakohuonepelin eteneminen ja vastaukset

Ensin tehtävänä on etsiä resepti. Velhokissa antaa tähän vinkkiä juuri katsotulla videolla. Lisäksi taululla on kuva Bluebotista ja puhekuplasta, jossa lukee ”Aloita minusta!” Jokaisella tuvalla on oma tehtäväruutunsa ja Bluebot-robottinsa luokan lattialla. Yhdessä ruudussa on taiteltu paperi. Sen sisällä olevaa tekstiä ei näe. Tuvan tehtävänä on ohjelmoida BlueBot tiettyyn ruutuun. Kun pääsee ruutuun, paperin saa avata. Ruudussa on liemireseptin ainesluettelo. Listan löytymisen jälkeen tehtävänä on etsiä mainitut ainekset.



Kuva 7: Ruudusta saatu aineslista

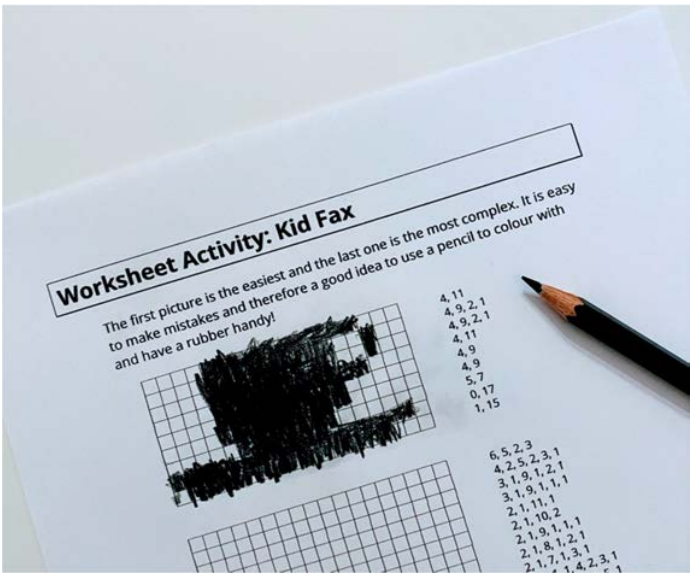
Liemen ainekset löytyvät luokkahuoneesta seuraavasti:

Liskonaivojauhot ovat opettajan pöydän takana hyllyllä. Hyllyllä on viisi läpinäkyvää lasipurkkia. Säännön mukaan alueelle ei kuitenkaan saa itse mennä.

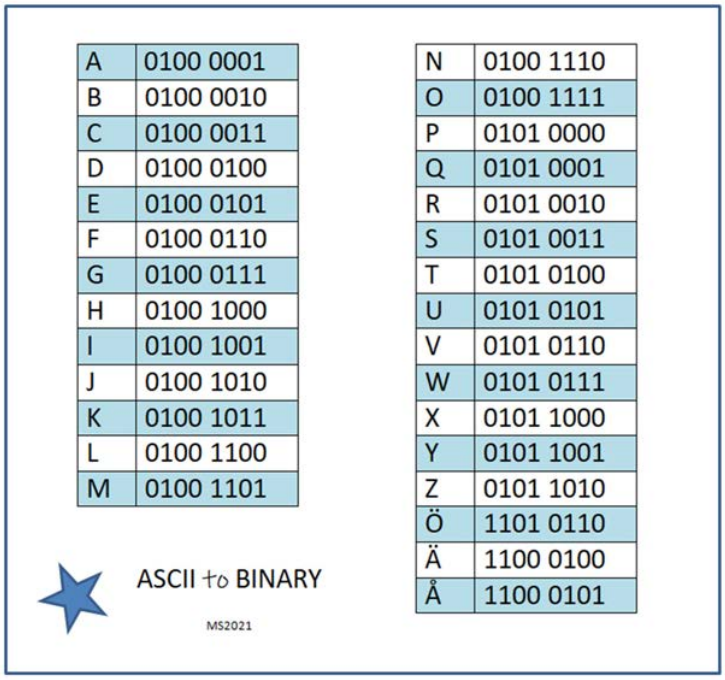
Luokan yhdelle pöydälle on laitettu taskulamppuja ja purkkien kuvia. Tehtävänä on katsoa purkkikuvia taskulampun valoa vasten. Näin tehden yhdestä purkkikuvasta löytyy liskon kuva, toisesta teksti ”Liskonaivojauhot ovat open hyllyllä.” Ongelmana onkin se, miten jauhopurkin voi saada jos opettajan pöydän taakse ei saa itse mennä. Kolmannesta purkista löytyy teksti ”Ope auttaa ja hymyilee, kun sille kahvia tarjoilee!”

Kuva 8: Taskulampulla valaistut purkit

Luokassa on yhdellä pöydällä väritystehtäviä. Tarkoituksena on huomata alkaa tehdä pikseliväritystehtävää. Pikseliväritykseestä muodostuu kahvikuppi. Kun sen vie opettajalle, saa liskonaivojauhepurkin opettajan pöydän takaa. Sen lisäksi opettaja antaa binääriavaimen. Binääriavainta tarvitaan jatkossa.



Kuva 9: Oppilaan ratkaisema pikseliväritystehtävä



A	0100 0001
B	0100 0010
C	0100 0011
D	0100 0100
E	0100 0101
F	0100 0110
G	0100 0111
H	0100 1000
I	0100 1001
J	0100 1010
K	0100 1011
L	0100 1100
M	0100 1101

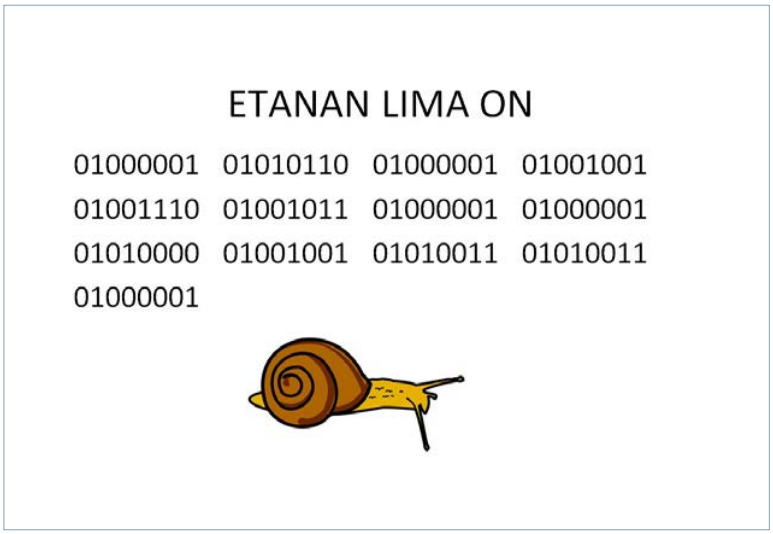
N	0100 1110
O	0100 1111
P	0101 0000
Q	0101 0001
R	0101 0010
S	0101 0011
T	0101 0100
U	0101 0101
V	0101 0110
W	0101 0111
X	0101 1000
Y	0101 1001
Z	0101 1010
Ö	1101 0110
Ä	1100 0100
Å	1100 0101

ASCII to BINARY

MS2021

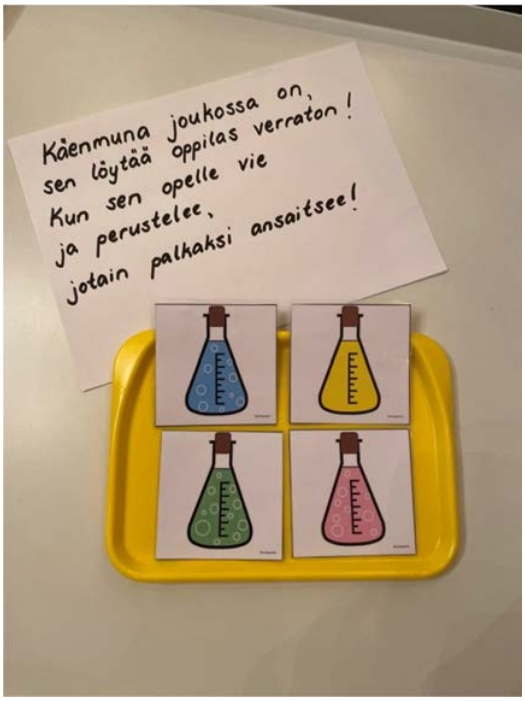
Kuva 10: Binääriavain

Etanan lima löytyy taulun alapuolelta, avainkaapeista. Etanan liman sijainnin saa selville taululla olevan etanavihjeen avulla ja binääriavainta käyttämällä. Avainkaapit ovat kuitenkin lukossa. Tarvitaan siis avain. Avain on perusmuotoinen tavallinen avain.



Kuva 11: Taululla oleva etanavihje. (Vastaus: Etanan lima on avainkaapissa)

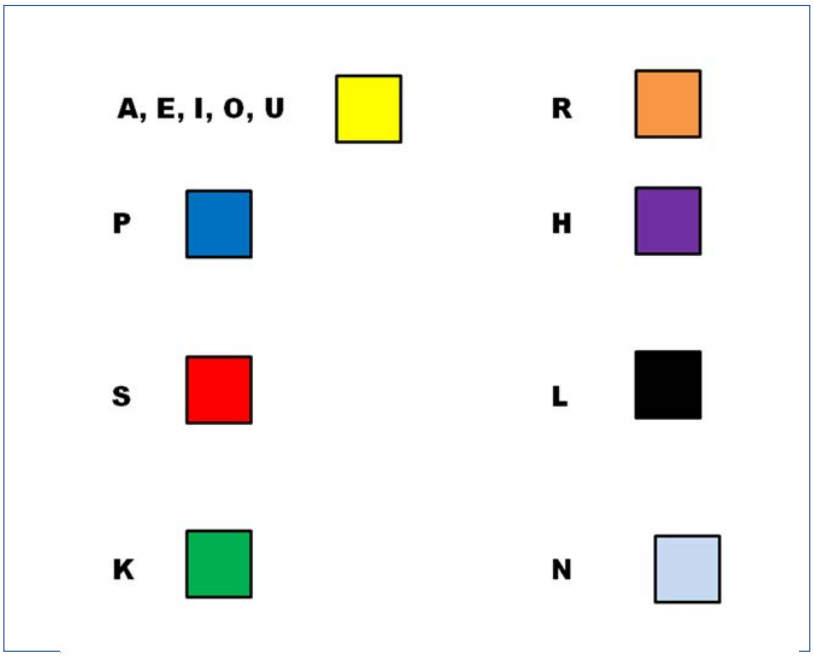
Luokan pöydille on aseteltu loogisten liemipullojen ryhmiä. Näitä tehtäviä on yhteensä neljä, jokaiselle tuvalle omansa. Jokaisen tuvan oma tehtävä on merkitty heidän tuvan värillä (esimerkiksi keltaisen tarjottimen tehtävä on Puuskupuhille). Liemipullojen yhteydessä on myös lappu jossa lukee: ”Käenmuna joukossa on, sen löytää oppilas verraton! Kun sen opelle vie ja perustelee, jotain palkaksi ansaitsee!” Tupa saa avaimen, kun palauttaa tietyn joukkoon kuulumattoman kortin opelle ja osaa perustella miksi se ei kuulu joukkoon. Avain sopii taulun alapuolella olevaan kaappiin. Tupa saa etanan liman avaamalla kaapin.



Kuva 12: Puuskupuhin tehtävä: Käenmuna on oikean yläreunan kortti. Muissa pulloissa on kuplia mutta siinä ei.

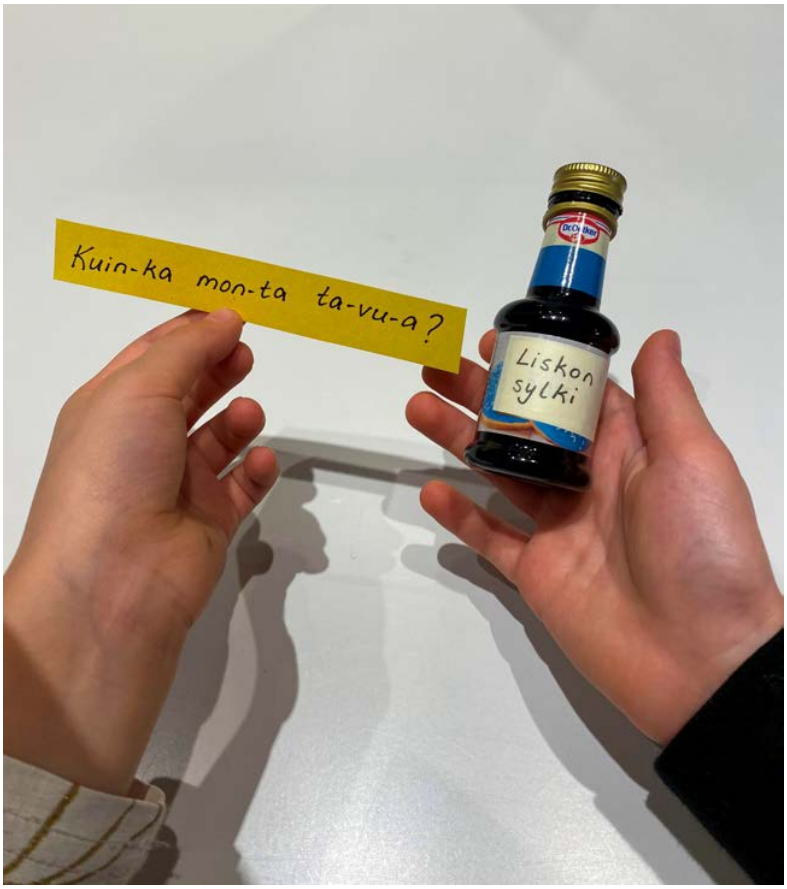
Kuva 13: Avainkaapista löytynyt etanan lima ja uusi vihje

Peikon räkä tai liskon sylki saadaan rakentamalla torni oikeanvärisistä duplopalikoista. Etanan limapurkin lisäksi avainkaapista löytyy myös lappu, jossa lukee: ”Rakenna, rakenna, tuvan nimi duploista rakenna! Ja torni taululle palauta!” Luokassa on iso laatikollinen eri värisiä duplopalikoita. Taululla on avain duplorakentamiseen, eli kerrottu mikä kirjain vastaa mitäkin duplopalikan väriä. Tarkoituksena on rakentaa oman tuvan nimi. Taululle on rajattu taulutussilla alue torneja varten, ja kirjoitettu ”tornien palautuspaikka”. Kun tupa palauttaa torninsa taululle, opettaja antaa tuvalle joko peikon räkää tai liskon sylkeä. Tupa saa valita kumpaa ottaa. Peikon räkä on punaista elintarvikeväriä, liskon sylki sinistä. Pikkupullossa on myös kiinni seuraava vihje.



Kuva 14: Duplorakentamisen avain

Kuva 15: Koodin mukaisesti rakennettu PUUSKUPUH-torni



Kuva 16: Puuskupuh-tupa on saanut liskon sylki –pullon ja ja uuden heidän keltaisselle paperille kirjoitetun vihjeen

Peikon rään tai liskon syljen lisäksi oppilaat saivat seuraavan vihjeen. Pullossa on kiinni myös paperi, jossa lukee kysymys. Papereita on neljä, jokaiselle tuvalle omansa. Niissä lukee:

Rohkelikon punainen paperi: ”Kuinka monta kirjainta on ensimmäisessä sanassa?”

Korpinkynnen sininen paperi: ”Kuinka monta kirjainta on toisessa sanassa?”

Puuskupuhin keltainen paperi: ”Kuinka monta tavua?”

Luihuisen vihreä paperi: ”Kuinka monta sanaa?”

Tehtävänä on ensin huomata, että laskeminen tapahtuu taululla olevasta ”Pöl-lö len-tää.” –virkkeestä.

Yksisarvisen kyyneleet eli viimeinen ainesosa löytyy kaapista, joka on suljettu pyöränlukolla. Avaamiseen tarvitaan jokaiselta tuvalta yksi numero. Lukon saa auki nelinumeroisella koodilla. Tähän saakka tuvat ovat toimineet itsenäisesti, mutta viimeistään tässä kohtaa tuvat tekevät yhteistyötä keskenään. Jos joku tupa ehtii saada tehtävät valmiiksi ennen muita, toisia saa toki auttaa. Opettaja voi ohjata tähän tehtävän loppupuolella. Opettaja voi taitavasti ohjailla oppilaita niin, että toisten auttaminen tapahtuu liikaa toisille paljastamatta ja että kaikki saisivat ratkoa tehtäviä itse. Yksisarvisen kyyneleet ovat kaikille yhteisessä pullossa. Opettaja voi ottaa pullon talteen ja liemenvalmistuksen vaiheessa kaataa jokaiselle tuvalle pienen määrän.

Kuva 16: Yksisarvisen kyyneleet löytyvät viimeisen nelinumeroisen lukon takaa

Pöl-lö len-tää.

Lukko aukeaa kun luvut järjestykseen paat.

Pienin on se joka aloittaa.

Tässä joka tuvan yhteistyötä tarvitaan!

Kuva 17: Luokan taululla oleva Pöllö lentää- virke ja vihje

Pöllövirkkeen vastaukset ovat seuraavat:

Kuinka monta kirjainta on ensimmäisessä sanassa? kirjainten määrä(5)
Kuinka monta kirjainta on toisessa sanassa? kirjainten määrä (6)
Kuinka monta tavua? tavujen määrä (4)
Kuinka monta sanaa? sanojen määrä (2)
lukon koodi (neljä numeroa pienimmästä aloittaen): 2456

Kun jokainen tupa on saanut kaikki ainekset kerättyä, opettaja ohjaa oppilaat liemenvalmistuspaikalle, luokan eteen pöydän ääreen. Oppilaat laittavat kaikki keräämänsä ainekset pöydälle. Ensin tulee lukea ohje huolellisesti. Sen jälkeen valmistetaan liemi. Kun kaikki muut ainekset on sekoitettu, opettaja kaataa jokaiselle tuvalle omaan kuppiin viimeisen ainesosan eli yksisarvisen kyyneliä. Yksisarvisen kyyneleet lisätään liemeen viimeiseksi.

LIEMIRESEPTI

1.Mittaa ja laita isoon kuppiin huolellisesti
5 teelusikallista liskonaivojauhoja
1 teelusikallinen etanan limaa
3 tippaa peikon räkää tai liskon sylkeä.
2.Sekoita.
3.Lisää lopuksi
3 ruokalusikallista yksisarvisen kyyneliä.

Lisätietona opettajalle:
liskonaivojauhoja eli soodaa
etanan limaa eli fairya
peikon räkää/liskon sylkeä eli elintarvikeväriä
yksisarvisen kyyneliä eli etikkaa

Kuva 18: Resepti liemen valmistukseen

Kuva 19: Valmista tuli!

Lähteet:

Opetushallitus 2014. Perusopetuksen opetussuunnitelman perusteet 2014. Helsinki. https://www.oph.fi/sites/default/files/documents/perusopetuksen_opetussuunnitelman_perusteet_2014.pdf (luettu 30.10.2021)

Ideoiden lainaaminen ja muokkaaminen omaan opetukseen:

Anniina Hakkaraisen OpenIdeat-blogi: Liemenvalmistusohje, jota muokkasin sekä nimien että määrien osalta. Muokkaamani versio on sopiva tehtäväksi isompaan pilttipurkkiin. http://openideat.blogspot.com/2021/02/ekaluokkalaiset-viistokujalla.html

Idea taskulampulla läpivalaisuun ja purkkeihin: Instagram _leikinlaillla_ -tililtä, ja hän oli edelleen lainannut idean Instagramista tililtä betzold.de

Idea loogiseen kokoelmaan Varga Nemenyi –koulutuksista

Pikseliväritystehtävä löytyy täältä: https://classic.csunplugged.org/documents/books/english/CSUnplugged_OS_2015_v3.1.pdf

Videotervehdyksen voi luoda Vokilla: https://www.voki.com/site/create

Tässä tuntisuunnitelmaraportissa käytetyt kuvat:
Etanan kuva
Sergio Palao / ARASAAC
Valkotaulun kuva
Sergio Palao / ARASAAC
https://kuvapankki.papunet.net/

BeeBot ja suuntien harjoittelu, 2.lk

5.10.2020

Tuntisuunnitelma

Algoritmisen ajattelun kehittäminen, Lumatikka koulutus

[redacted] 5.10.2020

Kohde ryhmä: 2-luokka (16 oppilaan ryhmät)

Ensimmäinen tunti

Ensimmäisellä tunnilla alustetaan hieman beebottien käyttöön. Ensimmäisenä keskustellaan suunnista. Opettaja piirtää taululle nuolien avulla suunnat eteen, taakse, käännös vasempaan- ja oikeaan samalla näyttäen ja selittäen, mitä käsitteet tarkoittavat. Sen jälkeen, pelataan “opettaja on robotti” leikkiä, jossa oppilaat vuorotellen saavat pyytää opettajaa tekemään asioita. Pyritään saamaan opettaja liikkumaan jonkun oppilaan viereen istumaan käskyillä eteen-taakse-käännös vasempaan tai oikeaan. Jos ohjeet ovat liian monimutkaisia, opettaja puistaa päätään ja jatkaa vasta sitten kun ohjeet ovat riittävän yksinkertaisia.

Riippuen siitä, miten kauan tässä on mennyt oppilaat saavat tehdä tätä pareittain niin, että koitetaan saada kaveri liikkumaan luokassa taululle ja kirjoittamaan oma nimensä. Tämä voidaan toteuttaa myös seuraavan tunnin aluksi tai joskus muulloin sopivana aikana.

Meidän koulussa sattumalta Ipadeissa oli valmiina Beebot peli, jossa hyvin käy selväksi beebottien toimintaperiaate. Pelissä beebot liikkuu erilaisilla alustoilla ja sille on ohjelmoitava komennot. Tähän peliin otan ensin pieni alustuksen ja oppilaat saavat pelata hetken ajan ja opetella beebottien toimintaperiaatetta ennen seuraavaa tuntia.

Toinen tunti

Toisella tunnilla otetaan sitten käyttöön itse Beebotit ja ensin esimerkillä näytetään kuinka beebot toimii. Olen katsonut itse valmiiksi ohjevideon youtubesta https://www.youtube.com/watch?v=Y6hhSNXXUOA ja ohjaan oppilaita käyttämään beebottia samalla tavalla, painottaen suunnat ja kääntymisen. Olen valmiiksi piirtänyt isolle värityspaperille 6x15cm+6x15cm cm neliöitä ja tehnyt näitä papereita 6kpl eli jokaiselle ryhmälle. Ensin teemme yhdessä testin jossa ohjelmoimme beebotin kulkemaan langan näyttämän reitin. Sitten oppilaat saavat ryhmässä oman beebotin (6kpl) ja ruudukolle laitetaan kukan kuva, johon beebotin pitää löytää. Olen leikannut erilaisian kuvia valmiiksi mihin pitää löytää ja mitä pitää väistää (kukka, traktori, baanaani, jne). Mikäli aikaa jää ja ohjelmointi on helppoa, voidaan reiteille asettaa jotain mihin beebot ei saa osua tai sen pitää löytää takaisin kotipesään. Aikaisempi osaaminen auttaa pääsemään nopeammin “vaikeampiin ratoihin” joita opettaja voi käydä tekemässä.

Aihepiirin valinta ja rajaus, työskentelyvälineet

Olen opettajana kielikylpykoulussa ja tämä toimii niin matematiikan tuntina, mutta ennenkaikkea kielikylpykielen tuntina, jossa opettelemme tärkeitä käsitteitä (suunnat). On myös mukava saada tehdä jotain uudenlaista oppilaiden kanssa, kun ryhmät eivät ole toteuttaneet tätä ykkösluokalla ja kun koulusta beebotit löytyvät. Beebottien mukana on kuulemma joskus ollut valmiina sellainen matto, mutta se on matkan varrella mennyt hukkaan. Siksi tein jokaiselle beebotille 6kpl omat alustat.

Oppimistavoitteet ja opetusmenetelmät

Oppimistavoitteina pitäisin käsitteiden oppimista ja ohjelmoinnin perustaitoja sekä “robottimaisen” kielen oppimista yksinkertaisten komentojen kautta. Myös ryhmäytyminen, parityöskentely ja ongelmanratkaisu ovat osa kokonaisuutta. Toivon, että nään oppilaiden onnistumisen Beebottien kanssa ja ehdottomasti odotan sitä, innostuvatko oppilaat tästä. Jatkamme työskentelyä Beebottien kanssa mikäli tämä motiovoi ja innostaa oppilaita. Opettajana seuraan ryhmien/parien työskentelyä ja arvioin myös käsitteiden oikeaoppista käyttämistä. Oppilaat saavat toivottavasti onnistumisen kokemuksia tunnin aikana joka on heille paras palaute!

Unplugged ja Lightbot, 2.lk

Sini Hostikka, CC BY-SA 4.0

5.5.2019

2. luokkalaisten algoritmiseen ajatteluun tutustumista

Ohjelmointitunnit 2. luokkalaisille (olen laaja-alainen erityisopettaja minulla ei ole omaa opetusryhmää, tunnit on toteutettu neljän oppilaan kanssa)

1. tunti

Paikkamerkit, 3 aluksi määrää kasvatetaan. Opettaja näyttää mallia astuu paikkamerkit tehden jokaisella merkillä jonkin liikkeen. Oppilaat toistavat liikkeet kävellessään paikkamerkkien päältä. 1. muutos lisätään paikkamerkkien lukumäärää. 2. muutos joka toinen tuplaa edellisen liikkeet ja joka toinen puolittaa ne. 3. muutos joka toinen tekee liikkeen peilikuvana/päinvastoin ja joka toinen palauttaa liikkeen alkuperäiseksi. 4. muutos jos on tyttö tekee peilikuvan/päinvastoin liikkeen, jos on poika tekee alkuperäisen liikkeen. (tehdään leikin aikana kortit for = toista, if= jos olet parillinen tuplaa, jos pariton puolita, if-else= jos tyttö, tee päinvastoin, muutoin tee alkuperäinen liike)

Piirrä ohjeen mukaan ruutupaperille, kulje viivoja pitkin. 1. muutos pura kaverin tekemä piirros toimintaohjeiksi nuoli vasemmalle, nuoli oikealle, nuoli ylös, nuoli alas.

Arvaa mikä pala otsallasi. Yksi leikkijöistä saa kruunun, johon laitetaan sinitarralla loogisten palojen yksi pala, hän arvuuttelee muilta kysyen palan eri ominaisuuksia ehdotellen, muut saavat vastata joko kyllä tai ei. Loogisten palojen ominaisuuksia: Reiällisyys reiätön-reiällinen, värit sin-pun-kelt-vihr, muoto kolmio-neliö-ympyrä, koko iso-pieni.

2. tunti

Aloitetaan tunti portinvartijat-leikillä leikinjohtaja-ope jakaa lapsia eri ominaisuuksien mukaan ja lapset pyrkivät arvaamaan mikä on sääntö (vaikeaa näin vähällä oppilasmäärällä). Pelataan Robogem-peliä. 1. muutos laitetaan kortteja riviin ja pyritään löytämään toistuva koodin pätkä, jonka voi vaihtaa funktio kortiksi. Tavoitellaan mahdollisimman pitkiä komentorivejä, joissa toistuisi pätkiä. Otetaan tabletit ja tutustutaan Lightbot-sovellukseen (meillä on vain lite-versio). Rakennetaan paikkamerkeistä omat Lightbot radat ja tehdään korteilla toimintakoodit niille. Vaihdetaan kaverin radalle.

Lähteet Varga nemenyi ja Ville- alakoulun ohjelmointi

Aarteenmetsästystä, 2.lk

22.2.2021

- Lähestyn ohjelmointia äidinkielessä puhumisen ja kuuntelemisen kautta, sillä ne ovat tärkeitä taitoja koululaisille
- Aihepiiriini liittyy ilmiönä koululaisen arkeen ja tunnilla toimimisen haasteisiin sekä itsensä ilmaiseimseen
- Oppimistavoitteina näissä ilmiössä on se, että oppilaan tulisi oppia kuuntelemaan ja toimimaan ohjeiden mukaan sekä ilmaisemaan itseään sanallisesti
- Ohjelmoinnin näkökulmasta tavoitteena on oppia ohjeiden noudattamista, kuten myös äidinkielen näkökulmasta. Äidinkieleen liittyen tavoitteena on myös oppia ilmaisemaan itseään
- Opettaja arvioi osaamista oppilaan tehtävien tekemisen pohjalta. Koska osaamistavoitteet menevät päällekkäin, arvioidaan osaamista kokonaisuutena
- Oppilas arvioi omaa osaamistaan peilaten tehtävissä onnistumiseen ja niistä tulevaan suoraan palautteeseen

- Oppikokonaisuuteen tarvitaan koulun jumppasalia, huiveja ja narua, penkejä jne
- Kokonaisuuteen lähdetään sillä, että oppilaiden kanssa keskustelaan luokassa millaisia erilaisia ohjeita he tietävät ja mitä taitoja he tarvitsevat erilaisten ohjeiden noudattamiseen. Oppilaita johdatellaan siihen, että ohjeita pitää kuunnella ja noudattaa tarkasti. Sitten keskustellaan siitä, miten itse annat ohjeita ja sitä harjoitellaan luokassa pareittaan niin, että pyydetään toista hakemaan jokin tavara ilman, että tavaran nimeä saa sanoa.
- Oppilaat motivoidaan tehtävään leikin varjolla ja oman hoksaamisen kautta
- Opintokokonaisuuden tehtävät tehdään pareittain ja niissä vaaditaan paljon yhteistyötä.
- Oppilas voi tuoda aiempaa osaamistaan esille niin, että hän osaa jo toimia tehtävissä pyydettyjen ohjeiden mukaisesti ja tuntee leikkien säännöt
- Oppikokonaisuuden jälkeen ohjelmointiin voidaan lähteä tutustumaan niin, että käytetään ohjeiden noudattamisen ja antamisen taitoja yksinkertaisissa tietokoneilla tehtävissä ohjelmointitehtävissä

1. oppitunti 45 min

· Käydään luokassa edellä esitetty alkukeskustelu ja ensimmäinen harjoitus

· Pelataan kapteeni käskee -peliä, opitaan ohjeiden kuuntelemista

· Mietitään yhdessä, miksi ohjeiden kuuntelu on välillä vaikeaa

2. oppitunti 45 min

· Siirrytään koulun jumppasaliin parijonossa, jonka liikkumista opettajaa ohjeistaa (hiivitään, laahustetaan, kävellään rauhallisesti yms.)

· Nopeasti kerrataan ohjeiden kuuntelun tärkeimmät asiat

· Pareittain harjoitellaan ohjeiden antamista: toisella on silmät sidottuna ja hänet pitää ohjata kävelemään parin luokse

· Opettaja tekee lattialle köydestä, penkeistä tms labyrintin. Oppilaat ohjaavat pareittain toisen kulkemaan labyrintin läpi silmät sidottuna

· Mietitään yhdessä ohjeiden antamista ja mikä siinä oli helpointa/vaikeinta

Kahden kertotaulua ja ohjelmointia, 2. lk

Hanne Kajasviita, CC BY-SA 4.0

20.9.2022

Kahden kertotaulua ja ohjelmointia 2-luokkalaisille

Hanne Kajasviita

Ohjelmoinnissa tarvitaan paljon matematiikkaa. Matematiikassa käskyt ovat yksinkertaisia ja ne suoritetaan laskusääntöjen mukaisesti. Samat säännöt pätevät myös ohjelmointiin, jossa ohjeet ovat yksiselitteisiä ja ne suoritetaan annetussa järjestyksessä. Kertotaulujen ymmärtäminen on matematiikan perustaitoja. Kertotaulujen osaaminen helpottaa myös ohjelmoinnin silmukoiden ymmärtämistä ja kirjoittamista.

Tunnin tavoitteet:
- Tutustua tai vahvistaa kahden kertotaulua.
- Tutustua ohjelmoinnissa käytettävään kieleen.
- Oppia kirjoittamaan lyhyitä käskyjä.

1. Tunnelmaan virittäytyminen
   - Luetaan Hello Ruby – Maailman paras koodisatukirja – kirjasta sivut 1–37.

   (Jos on käytettävissä useampi tunti, voidaan keskustella jokaisen luvun pääaiheesta)

2. Pohditaan yhdessä.
   - Millaisia käskyjä robotille täytyy antaa?
     (yksinkertaisia)
   - Miten ongelmia kannattaa lähteä ratkomaan?
     (pilkkomalla ongelma pieniin osiin, kokeilemalla ja keskustelemalla)
3. Esitellään Robotti x2
   - Robotti x2 ominaisuudet:
   - Yhden askeleen pituus on 2 ruutua.
   - Se kulkee ylös, alas, oikealle ja vasemmalle.
4. Pohditaan, millaisia ohjeita Robotti x2:lle voidaan antaa
   - Sen halutaan kulkeva 2 ruutua eteenpäin.
   - Sen halutaan kulkevan 4 ruutua eteenpäin.

Lasten ehdotukset voivat olla esim. kulje 4 ruutua eteenpäin. Opettaja kirjoittaa ehdotuksen taululle ja pohditaan, kuinka viestin voi kirjoittaa nuolilla, numeroilla ja kertomerkillä.

5. Tutkitaan ryhmässä monisteen avulla millaisella käskyllä Robotti x2 pääsee aarteen luo.
6. Kerätään yhdessä tulokset taululle ja huomataan niiden yhteys kahden kertotauluun.
7. Tehdään ohjelmointitehtävä moniste tai annetaan se läksyksi.
8. Tarkistetaan ohjelmointitehtävän vastaukset.

- Alaspäin eriytettävien on hyvä tehdä töitä ryhmässä, jossa ei ole nopeimmat ja ylöspäin eriytetyt. Mahdollisuuksien mukaan ohjaaja tai opettaja avustaa toimintaa.

- Ylöspäin eriytetyt voivat piirtää itse ruutupaperille reittejä ja antaa niitä toisilleen ratkottavaksi.

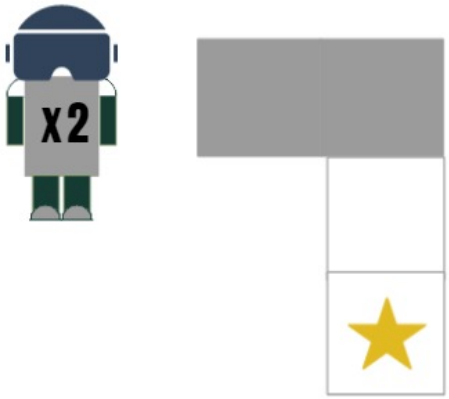
- Opettaja voi myös tehdä haastavampia ja pidempiä reittejä ratkottavaksi, jos työaika antaa myöden.

- Samaa ajatusta voidaan toteuttaa myös muilla kertotauluilla.

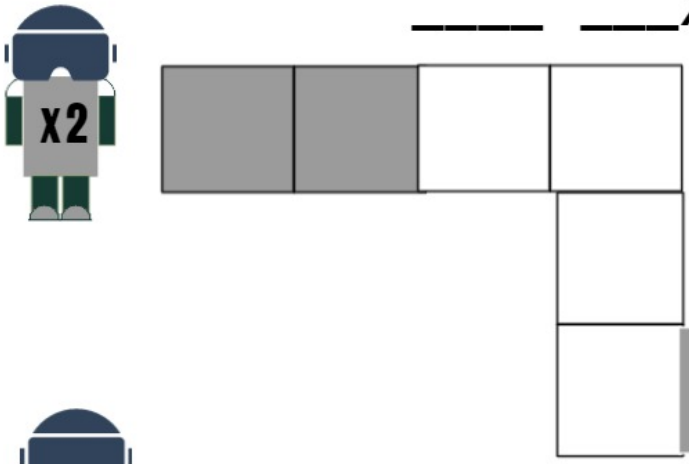
- Mikäli mahdollista, ohjelmointitaitojen ja ajatuksen vahvistamiseksi tehtäviä olisi hyvä päästä kokeilemaan esim. Scratch jr -ohjelmalla.

Monisteet löytyvät myös Freedistä https://www.freeed.com/articles/14943/2-kertotaulu-ja-ohjelmointi.

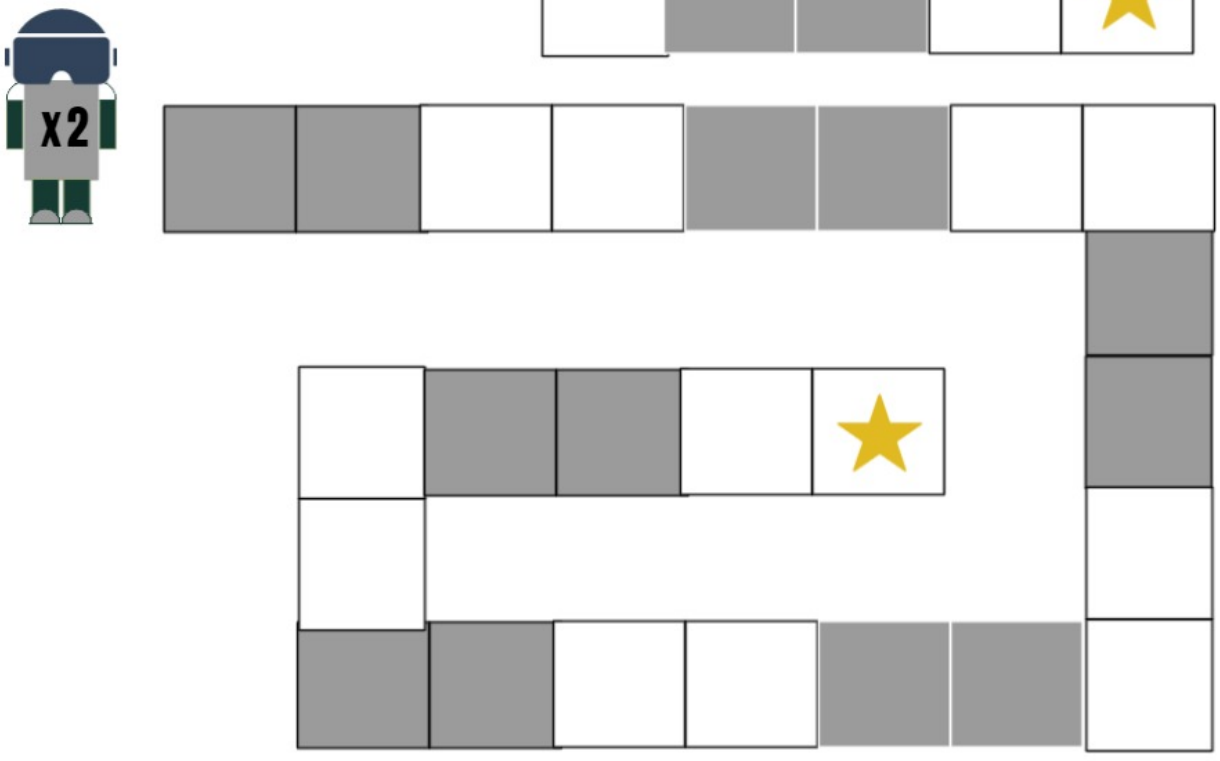
Tee ohjauskäsky Robotille, joka kulkee 2 ruutua yhdellä askeleella. Käytä suunnan ohjaukseen nuolia. → ↓ ← ↑



→ ___x2 ↓ ___x2



___ ___x2 ___ ___x2 ____ x2

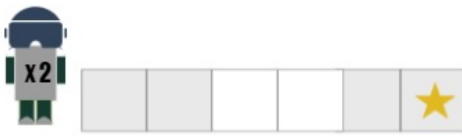


___x2 ___ ___x2 ___ ___x2 ___ ___x2

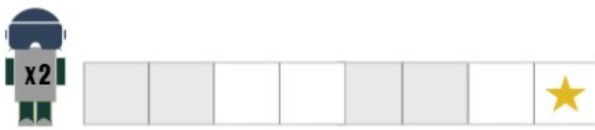
Robotti x2

____x 2 = _____

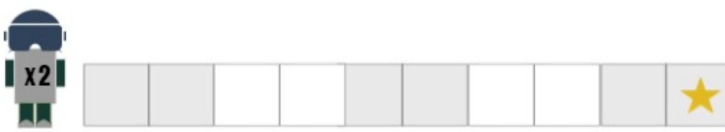
____x 2 = _____



____x 2 = _____



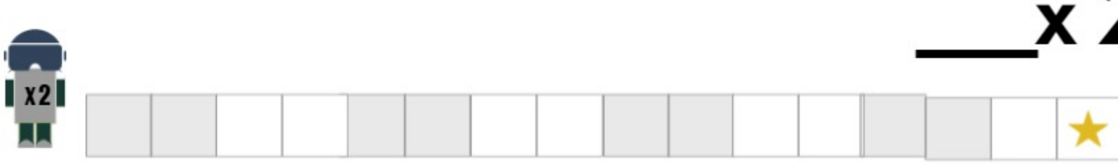
____x 2 = _____



____x 2 = _____

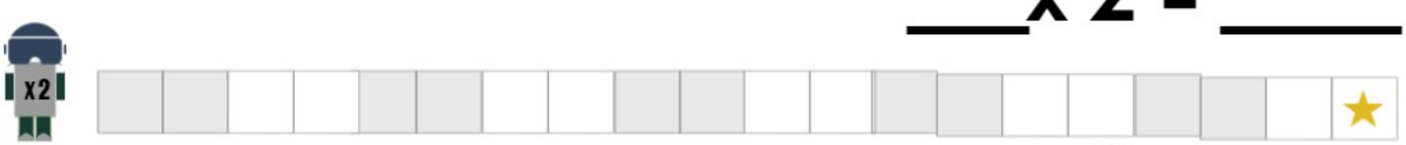
____x 2 = _____

____x 2 = _____



____x 2 = _____

____x 2 = _____



____x 2 = _____

Kertotaulun harjoittelua BeeBot-roboteilla, 2. lk

Nimetön,

10.10.2022

Halusin valita tunnin aiheeksi matematiikan, koska se on yleisimmin opettamani aine erityisopetuksessa. Oppilaat tarvitsevat paljon kertausta omassa luokassa käydyistä aiheista.

Alkuleikki tunnille on perinteinen kapteeni käskee mutta hieman erilaisilla ohjeilla. Leikitään käytävällä tai salissa niin, että jokaisella on tilaisuus liikkua. Kapteeni käskee ottamaan kaksi kertaa viisi askelta eteenpäin. Kapteeni käskee hyppäämään kaksi kertaa eteenpäin. Opettaja johtaa leikkiä. Leikin lopuksi puolet oppilaista jää paikoilleen ja puolet menee parin viereen. Tämä toinen oppilas yrittää nyt ohjelmoida oman parinsa yhdessä sovittuun lopetuspisteeseen kapteeni käskee leikin avulla. Oppimistavoitteena on ohjeen kuuntelu, ohjeen noudattaminen, kertolaskujen muistelu ja parityöskentely.

Seuraavaksi käytetään Beebotteja ja alustoja niihin. Tunnin alussa mietitään yhdessä, mitä robotilla voisi tehdä. Katsotaan aluksi yhdessä video https://youtu.be/Y6hhSNXXUOA.

Tunnin tarkoituksena on opetella ja kerrata kertotauluja, jotka pitäisi toisen luokan aikana oppia ulkoa. Lisäksi opetellaan beebotin käyttöä. Beebotille on taskumatto, johon olen laittanut taskuihin kertotaulun vastauksia ja värikoodeja. Esimerkiksi 25, punainen. Värejä on yhtä monta kuin kertotauluja opeteltavana. Yhteen mattoon mahtuu kahden kertotaulun vastaukset, joten lappuja voi vaihtaa tunnin aikana. Oppilas ottaa yhdestä korttipinosta laskun (esim 5x5) ja laskee sen. Sen jälkeen oppilas laittaa beebotin sovittuun lähtoruutuun ja ohjelmoi botin kulkemaan oikeaan vastausruutuun (25, punainen). Pari tarkistaa vastauksen kertolaskutaulukosta kirjasta. Tämän jälkeen oppilas ottaa uuden laskutehtävän oikean värisestä pinosta (punainen), laskee ja ohjelmoi. Tätä tehdään yhdessä parin kanssa niin kauan, että kortit loppuvat. Jos Beebot on hyvin hallussa, voi tehtävää eriyttää niin, että beebot pitää saada kulkemaan järjestyksessä kaikkien vastausten reitti.

Tunti pitäisi olla kaikille mieluisa, sillä olemme aiemmin kerran kokeilleet beebotteja ja ne tuntuvat olevan kaikkien suosikkeja. Toivottavasti ohjelmointi tuo kertotaulujen opetteluun myös puhtia. Oppimistavoitteena on kertotaulujen lisäksi ohjelmoinnin alkeet. Yhteistyötä tehdään tässä tehtävässä sellaisen parin kanssa, jonka opetta on valinnut niin kuin aina luokassani. Yhdessä tekemällä syntyy hyvää matikkapuhetta.

Tunnin jälkeen oppilaat ovat valmiina seuraavaan tuntiin, jossa harjoitellaan Scratch Juniorin käyttöä. He ovat saaneet konkreettista harjoitusta nuolikäskyjen käyttämisestä ja voivat siirtyä tekemään komentoja ipadillä.

Ohjelmointi ja käsityöt: sinivuokkoliinan ompeleminen, alkuopetus

Nimetön,

12.4.2021

Tuntisuunnitelma: sinivuokkoliinan ompelu

Toteutan tämän tuntisuunnitelman käsityötunneilla. Valmistamme äitienpäivälahjaksi säkkikankaisen pikku liinan, johon ompelemme huopakankaasta tehtyjä sinivuokkoja. Käsityötä on kaksi oppituntia (2x45min) peräkkäin ja näiden kahden tunnin aikana on tarkoitus ommella kukat liinaa varten. Työtä jatketaan myöhemmin käsityötunneilla.

Sinivuokon kukat ommellaan kuudesta terälehtipalasta ja kuudesta helmestä. Työ aloitetaan kukkien tekemisestä ja jokainen oppilas voi tehdä niitä eri määrän. Tavoitteena on valmistaa 1-6 kukkaa per oppilas.

Kukkaa ommeltaessa huopakankaasta leikatut terälehdet ja helmet levitetään pöydälle ompelujärjestykseen. Ensin terälehti, sitten helmi, terälehti, helmi jne. Viimeisenä on helmi.

Tässä tehtävässä toimitaan tarkasti ohjeen mukaan ja noudatetaan ohjeen mukaista koodia. Tavoitteena on opetella toistamaan samanlaista ja yksinkertaista koodia: terälehti, helmi, terälehti jne.

Tavoitteena on harjoitella ohjelmointitaitoja:

- Opetella noudattamaan ohjetta.
- Opetella seuraamaan annettua koodia.

Tavoitteena on harjoitella ompelutaitoja:

- Opetella pujottamaan lanka neulan silmään ja tekemään solmu langan päähän.
- Opetella ompelemaan terälehdet kiinni “alas-ylös”-pistoilla ja pujottaa helmet terälehtien väliin.
- Opetella päättelemään ompelutyö lukkopistoilla.

Opettaja ja ohjaaja arvioivat oppilaan osaamista työskentelyn lomassa. Ohjelmointia ja ompelua arvioidaan kokonaisuutena. Jos koodi ei ole mennyt ohjeen mukaan, esim. helmi on jäänyt välistä, voidaan koodi korjata purkamalla työtä tarvittava määrä. Oppilas arvioi omaa osaamistaan vuorovaikutuksessa opettajan ja ohjaajan kanssa. Jos oppilas valmistaa useampia kukkia, hän voi verrata niiden onnistumista, miettimällä, mikä onnistui parhaiten ja miksi.

Työskentelyssä työvälineenä on terävä neula ja tarvikkeina huopakangasta, helmiä, lankaa ja säkkikangasta. Tehtävien ohjeistus käydään läpi opettajajohtoisesti. Opettaja kertoo työn etenemisestä ja kirjoittaa lyhyet ohjeet piirroksineen luokan taululle.

Oppilaita motivoi työn lopputulos, sillä se annetaan äidille äitienpäivälahjaksi. Työ on myös melko yksinkertainen ja kun oppii tekemään yhden kukan, onnistuu seuraavan tekeminen jo melko itsenäisesti.

Oppilaat tekevät luokassa yhteistyötä mielellään ja hyvin omatoimisestikin. Erityisesti ne oppilaat, jotka oppivat työn tekemisen nopeasti, auttavat mielellään muita silloin, jos opettaja tai ohjaaja eivät ehdi.

Matematiikan tunneilla olemme tehneet useita ohjelmointiin liittyviä kooditehtäviä ja opettaja kehottaa tunnin alussa muistelemaan näitä tehtäviä. Tätä tehtävää verrataan matematiikan kooditehtäviin ja opettaja käyttää puheessaan samoja termejä kuin matematiikan tunnilla.

Opettaja voi edesauttaa opitun asian yhdistämistä muihin ohjelmointi tai algoritmista ajattelua vaativiin tilanteisiin pyytämällä oppilaita pohtimaan, mitä samanlaista tilanteissa on ollut.

Lähteenä on kollegalta saatu idea sinivuokkojen tekemiseen. Idean alkuperäislähde ei ole tiedossa.

Ohjelmointi ja käsityöt: paperilennokki, helminauha ja kirjonta, alkuopetus

22.1.2021

Tuntisuunnitelma: ohjelmointia käsitöissä

Aihepiirin valinta ja rajaus

Tässä tuntisuunnitelmassa ohjelmointia sovelletaan käsityöhön, koska etenkin alakoulussa käsityön ja ohjelmoinnin yhdistäminen on helppoa ja hauskaa. Käsityön ja ohjelmoinnin yhdistäminen myös avartaa ajattelemaan ohjelmointina muutakin, kuin tietokoneella tai matematiikan tunnilla tapahtuvaa ohjelmointia.

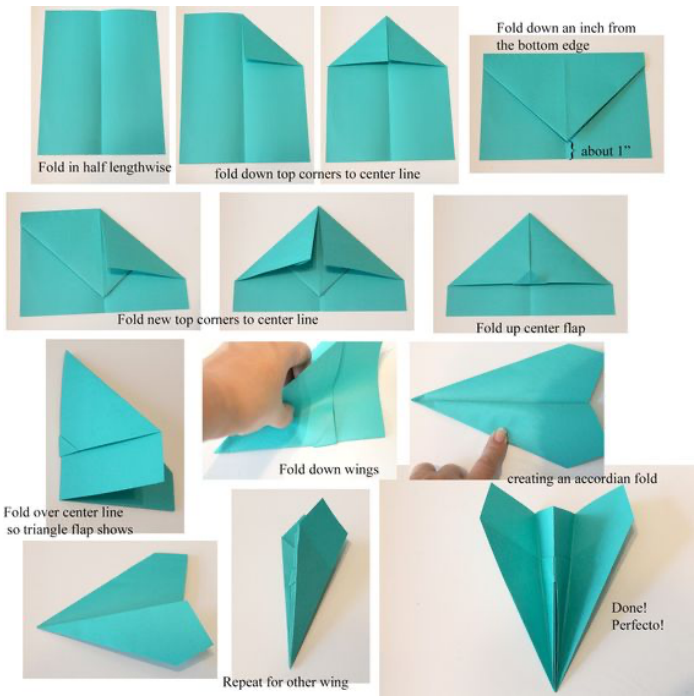
Tämän tuntisuunnitelman toteuttaminen sopii taidollisesti jo alkuopetukseen, mikäli lapset ovat jo jonkin kerran tutustuneet ohjelmointiin ja ohjelmoinnin peruskäsitteet ovat hallussa. Tehtävät ovat samaan aikaan ohjelmoinnilliseen ajatteluun johdattelevia ja että algoritmista ajattelua soveltavia.

Tavoitteet ja arviointi

Tavoitteena on ohjelmoinnillisen ajattelun kehittyminen ja ohjelmoinnin opiskelun hyödyn ymmärtäminen opiskelemalla sitä käsitöiden kautta. Tavoitteena on myös uusien oivallusten syntyminen ja konkretian avulla ohjelmoinnin tuominen lasten arkeen. Arvioinnissa painotetaan oppilaan kykyä soveltaa algoritmista ajattelua ja ohjelmointiin liittyviä käsitteitä käsitöiden suunnitteluvaiheessa. Arviointi tapahtuu formatiivisena arviointina oppilaan ja opettajan välisen keskustelun sekä opettajan havainnoinnin kautta.

Ensimmäinen oppitunti

Ensimmäisellä tunnilla johdannoksi otetaan **paperilennokin** tekeminen ohjeen mukaan. Ideana on herätellä lapset oivaltamaan, että mikä tahansa vaiheittain etenevä ohje on itse asiassa koodi, jota noudattamalla päästään haluttuun lopputulokseen.



Kuva: Pinterest

Tämän jälkeen tehdään **helminauhatyö**. Oppilaat suunnittelevat ensin paperille/vihkoon koodin, jolla saadaan toivottu lopputulos, eli missä järjestyksessä helmet pujotetaan helminauhaan. Tässä voidaan käyttää hyväksi yksinkertaisia ohjelmointiin liittyviä komentoja, kuten silmukoita tai ehtolauseita.

Toinen oppitunti

Toisella oppitunnilla oppilaat aloittavat **kirjontatyön vohvelikankaalle**. Opettaja voi antaa aluksi muutaman kuvion malliksi minkä jälkeen oppilaat voivat lähteä itse ”koodaamaan” erilaisia toistokuvioita, eli suunnittelemaan omia kuvioita. Koska kyseessä on käsityö, niin luonnollisesti tätä työtä jatketaan useamman seuraavan tunnin ajan.

Kuva: Pinterest

Ohjelmoinnin alkeet pistetyöskentelynä, alkuopetus

1.4.2021

Pistetyöskentely alkuopetukseen

Tuntisuunnitelma on tehty 1.-2.luokkalaisille matematiikan ohjelmointijaksolle. Tarkoituksena tutustuttaa oppilaita ohjelmoinnin alkeisiin pistetyöskentelyn avulla. Tehtävät ovat sellaisia, että oppilaat pystyvät niitä keskenään tekemään lyhyen ohjeistuksen avulla ja opettajalta vapautuu aikaa kiertää arvioimassa ja tarkkailemassa oppilaita.

Tavoitteena on tutustua ohjelmoinnin osa-alueisiin (ongelmanratkaisu, selkeä ohjeenanto, Beepot-robotin ohjelmointi). Etukäteisosaamista näistä ei tarvitse olla ja kaikkia osioita on tarpeen harjoitella vielä jatkossakin. Ennen kaikkea tavoitteena innostaa lapsia ohjelmoinnin pariin. Halutessaan liittää aiheeseen itsearviointia, voi opettaja tehdä oppilaille esimerkiksi ohjelmointipassit, joihin voi merkitä harjoittelemansa osa-alueen.

Kohderyhmä: 1.-2. luokkalaiset

Kesto: 2 oppituntia (2 x45 min, alussa ohjeistus pisteiden toiminnasta n.10 min, sen jälkeen toiminta n.15-20 min/piste n.4 hengen pienryhmissä)

Tavoite: Tutustua ohjelmoinnin alkeisiin eri tavoin

1. Selkeän yksinkertaisen ohjeen antaminen

Tarvikkeet: Multilink-palikoita

Ohjeet: Parin kanssa istutaan selät vastakkain ja molemmilla edessä Multilink-palikoita. Toinen antaa ohjeita miten rakennetaan ja rakentaa samalla myös itse ja toinen rakentaa ohjeiden mukaisesti rakennelman. Verrataan lopputuloksia. Tehtävä onnistuu myös kolmistaan siten, että yksi antaa ohjeita ja kaksi rakentaa.

2. Beepot Suomen kartalla

Tarvikkeet:

- Suuri Suomen kartta, jonka päälle pleksistä tehty ruudukko, jolla Beepot voi liikkua

- Beepot-robotti

- puupalikka

Ohjeet: Valitaan aloituspisteeksi esim.oma paikkakunta. Yksi ryhmästä päättää kohteen ja Beepotia ryhdytään ohjelmoimaan kohteeseen. Apuna puupalikka esim.Huojuva torni-pelistä, jolla liikutaan toivottua reittiä edeltä, jotta osataan miettiä, missä vaiheessa tarvitsee ohjelmoida käännös ja kuinka monta ruutua eteenpäin voi mennä suoraan. Vaihdetaan rooleja, siten että kaikki saavat toimia Beepotin ohjelmoijana. Mikäli mukana

on lukutaidottomia lapsia, on tärkeää huomioida, että ryhmässä on lukutaitoisia parina työskentelemässä.

3. Ihmisrobotin ohjaus

Työskennellään ryhmässä, yksi on robotti, toinen antaa ohjeita robotille, jonka pitää toimia ohjeiden mukaan. Loput leikkijät tarkkailevat, ettei robotti tee mitään ylimääräistä ja tottelee ohjeita oikein. Vaihdetaan rooleja siten, että kaikki ehtivät olla robotteja ja ohjeen antajia vuorollaan.

4. Tulitikkutehtävät (ongelmanratkaisun harjoittelu)

Tarvikkeet: tulitikkuja, tulostettuja tulitikkutehtäviä

Tulitikkutehtäviä on olemassa paljon valmiina ja suurin osa suosittelee tehtäviä 3.luokasta ylöspäin. Alkuopetuksessa tehtäviä voi hyvin tehdä pareittain tai ryhmässä, yleensä onnistuu.

Tehtäviä voi tulostaa esimerkiksi seuraavilta sivuilta

https://ryhmarenki.fi/wp-content/uploads/2018/04/Tulitikkupahkinat-1.pdf

https://ryhmarenki.fi/wp-content/uploads/2018/02/Tulitikkupahkinat_2.pdf

https://ouluma.fi/wp-content/uploads/2013/02/Tikkuteht%C3%A4vi%C3%A4-2-ratkaisuineen-2013.pdf

https://blogs.helsinki.fi/summamutikka/files/2014/07/Tulitikkuteht%C3%A4vi%C3%A4.pdf

LÄHTEET

INNOKAS-sivuston materiaalit 2021, Helsingin yliopisto. Saatavilla osoitteesta www.innokas.fi

Opetushallinto. Perusopetuksen opetussuunnitelman perusteet 2014.

Salainen polku: Ohjelmoinnin alkeita toiminnallisesti ilman tietokoneita, alkuopetus

Nimetön, CC BY-SA 4.0

18.4.2021

-Salainen polku-

Ohjelmoinnin alkeita toiminnallisesti ilman tietokoneita

Tämä kahden tunnin kokonaisuus sopii alkuopetukseen johdannoksi koodaamisen maailmaan. Opetuskokeilussa ryhmänä toimi kakkosluokkalaiset, joiden ryhmää opetan neljä tuntia viikossa. Testasin tätä tosin myös omalle luokalleni ja neljäsluokkalaiset innostuivat tästä ehkä jopa enempi. Varmasti tämä toimisi myös eskari-ikäisillä. Teimme tehtävän matematiikan oppitunneilla.

Kerroin, että meidän tulisi päästä salaisen polun kautta salaiseen maailmaan, jossa meitä odottaisi lisätehtäviä. Reitti on vaarallinen ja sen kulkemisessa on oltava todella tarkkana. Yksikin harha-askel saattaa olla turmioksi tai vähintään saattaa reitin alkuun.

Olin askarrellut teipin avulla luokan lattiaan ruudukon. Tähän ns. sähkömiehen teippi toimii paremmin kuin maalarinteippi, ainakin jos ruudukon on tarkoitus olla vähänkin pidempään paikoillaan. Maalarinteipin liimat saattavat jämähtää lattiaan. Ruudukko oli neljä ruutua x kahdeksan ruutua. Reitin aloitus oli toisessa neljän ruudun sivussa. Tarkoitus on siis päästä ruutuja kulkemalla salaisen polun reitti turvallisesti toiseen päähän. Reitti on piirretty etukäteen ruutupaperille ja vain siis reitin suunnittelija tietää oikean reitin. Ensimmäinen oppilas astuu summamutikassa johonkin ruutuun. Reitin suunnittelija kertoo, oliko ruutu oikein vai ei. Oikeasta ruudusta saa jatkaa reittiään seuraavaan valitsemaansa ruutuun. Väärästä valinnasta putoaa pois ruudukolta ja palaa takaisin alkuun. Muiden tehtävä on seurata yritystä silmä kovana ja olla omalla vuorollaan siis tietenkään astumatta samoihin ansaruutuihin kuin edeltänyt oppilas. Jatketaan niin kauan, kunnes kaikki ovat päässeet turvallisesti salaisen polun reitin ruudukon läpi.

Tässä harjoituksessa arvioidaan vain oppilaan oma arvio onnistumisesta: onnistuinko selvittämään salaisen reitin ja pääsinkö lopulta toiselle puolelle? Jokainen pääsi eli kaikki onnistuivat tavoitteessa loistavasti! Tässä ei ollut kyse nopeudesta, joka kerrottiin oppilaille ääneenkin. He hoksasivat myös itse, että ensimmäinen urakka oli vaativin. Toisella oli jo yhden ruudun verran helpompaa, sillä hän tiesi, mihin ei ainakaan tule astua. Tässä tuli harjoitettua myös muistia. Kun ensimmäinen oli onnistuneesti päässyt salaisen polun läpi, ei toisilla välttämättä ollutkaan muistissa tarkasti kuljettu reitti. Kun suurin osa oppilaista oli päässyt reitin läpi, sai halutessaan kysyä neuvoa jo läpäisseiltä. Ei ollut tarkoitus, että kukaan jää yksin reittiä junnaamaan vaan hauskuus ja onnistumisen kokemus.

Kokeilussa minulla oli ensimmäisellä tunnilla puolikkaan ryhmän tunti – onneksi. Oppilaat olivat innoissaan tehtävästä, mutta kovin malttamattomia. Tiputtuaan olisi ollut hankalaa odottaa uudelleen omaa vuoroaan, jos edellä olisi ollut ison ryhmän kaikki oppilaat. Vaihtoehtoisesti voisi ohjeistaa osan luokasta tekemään itsenäisesti tehtäviä ja vain osa tulisi kerrallaan yrittämään salaisen polun suorittamista. Oppilaista jokainen olisi halunnut suunnitella oman salaisen polkunsa ja antaa muille palautetta oikeasta ja väärästä ruutuvalinnasta. Tätä perustehtävää on siis myös ehdottomasti jatkettava lisätehtävänä toisella kertaa, jotta jokainen pääsee vuorollaan kartan piirtäjäksi.

Seuraavalla tunnilla ei päässytkään itse polulle vaan tehtävänä oli saada kaveri turvallisesti polku päästä päähän. Kaveri piti siis koodata astumaan askel eteenpäin / askel vasemmalle / askel oikealle. Me emme ottaneet reittimahdollisuudeksi siirtyä ruudun kulmasta vinoittain toiseen ruutuun, sekin toki olisi mahdollista. Lisäksi voisi ohjelmoida kaverin myös niin, että se ei ottaisikaan sivuaskeleita vaan pitäisi aina ensin kääntyä kasvot etenemissuuntaan.

Huomasin, että oppilaat pyrkivät tiedostamattaan lukemaan toisen ajatuksia ja toimimaan, vaikka käskyä ei olisi tullut tai käsky olisi ollut puutteellinen. Tarvittiin siis alkuun myös operobotti koodattavaksi, joka ei osannutkaan toimia lainkaan ilman ohjeita tai toimi juuri ohjeiden mukaan, mutta väärin. Pikkuhiljaa oppilaiden ohjeet täsmentyivät.

Kun oli sekä saanut olla itse koodattavana että koodata toisen reitin läpi, keskustelimme ja vertailimme tätä tuntia ja edellistä tuntia. Oliko helpompaa suorittaa reitti itse, kulkea se toisen koodaamana vai koodata toinen reitin läpi? Miksi? Syntyi monenlaista hyvää pohdintaa ja aiheeseen ei syntynyt lopullista vastausta. Osa esimerkiksi koki, että oli kiva olla toisen koodattavana, sillä ”ei tarvinnut ajatella itse mitään”. Osa taas selkeästi piti itse reitin pähkäilystä. Osan mielestä parasta oli olla kartanpiirtäjä ja suunnitella reitti ruutupaperille.

Tämän kokeilun luonteva jatko voisi olla Beebot-robottien koodaaminen. Lisäksi aihetta voisi jatkotyöstää esimerkiksi kuvataiteen ja suomen kielen tunneilla: Millaiselta salaisen polun päässä näytti? Keitä siellä asuu? Tapasitko ketään? Ymmärsittekö toisianne? Mitä teit siellä? Kirjoita ja piirrä.

https://koodiaapinen.fi/

https://mooc.helsinki.fi/pluginfile.php/152755/mod_resource/content/11/ohjelmointi-ja-internet-harjoituksia.pdf

https://www.oph.fi/fi/koulutus-ja-tutkinnot/perusopetuksen-opetussuunnitelman-perusteet

Ohjelmointia nuolikorteilla, alkuopetus

10.11.2022

Tein tuntisuunnitelman alkuopetukseen. Tehtävässä ei tarvitse osata lukea ja tavoitteena on tutustua ohjelmointiin ja oppia itse ohjelmoimaan.

Valmistelut:

Laminoituja nuolikortteja valmiiksi taululle. Tehtävään tarvitaan suoria nuolia sekä oikealle ja vasemmalle kääntyviä nuolia.

Tunnin aloitus ja kiinnostumisen herättäminen:

Tehdään oppilaiden kanssa opettajajohtoisesti luokkaan labyrinttimainen rata. Kun rata on valmis, asettuu opettaja radan toiseen päähän ja antaa oppilaille tehtäväksi ohjata hänet radan läpi käyttämällä nuolia. Viittaamalla saa vuoron ja voi käydä asettamassa taululle nuolen siten, kun haluaisi opettajan liikkuvan. Opettaja ottaa aina askeleen nuolen osoittamaan suuntaan.

Työskentely:

Seuraavaksi oppilaat voivat muuttaa rataa ja joku oppilaista valitaan suorittamaan rataa. Taas oppilaat saavat ohjelmoida toverinsa radan läpi. Tämä voidaan tehdä muutaman kerran.

Pulpetit laitetaan paikoilleen ja oppilaat saavat parin kanssa suunnitella omia liikekortteja. Voisiko labyrintissä olla erilaisia tehtäviä suoritettavana? Vaikkapa pyörähdä ympäri, hyppää yhdellä jalalla tai taputa. Oppilaat saavat keksiä omia symboleja erilaisille liikkeille.

Koonti:

Lopuksi voidaan katsoa yhdessä, mitä liikkeitä oppilaat ovat keksineet ja mitä symboleita niitä kuvaamaan.

Käydään myös läpi, mitä juuri tehtiin. Ohjelmointia, jolla annettiin erilaisia käskyjä radasta suoriutumiseen. Opettaja kertoo, että ohjelmointia käytetään annettaessa tietokoneille erilaisia tehtäviä ja olemassa on erilaisia ohjelmointikieliä.

Jatko:

Oppilaiden kehittämiä symboleita voidaan käyttää esimerkiksi liikuntatunnilla. Oppilaat voivat ryhmissä tai pareittain aina suunnitella radan ja muut oppilaat suorittavat sen.

Voidaan myös siirtyä ohjelmoimaan esim. Scratch juniorin avulla ja syventää opittua.

Tämä tehtävä sopii matematiikan ja liikunnan tunneille. Kun ohjelmointiin on tutustuttu ensin matematiikan tunnilla, voidaan samaa asiaa harjoitella liikunnallisesti. Näin saadaan myös useampia toistoja ennen kuin siirrytään ohjelmoimaan esimerkiksi Scratch Jr ympäristöön. Tehtävä tutustuttaa oppilaat ohjelmoinnin perusteisiin ja vastaa laaja-alaisen oppimisen tavoitteeseen L5 Tieto ja viestintätekniikan osaaminen.

Tavoitteet:

Ohjelmointi:

Tutustua ohjelmoinnin perusteisiin.

Matematiikka:

T1 tukea oppilaan innostusta ja kiinnostusta matematiikkaa kohtaan sekä myönteisen minäkuvan ja itseluottamuksen kehittymistä

T4 ohjata oppilasta kehittämään päättely- ja ongelmanratkaisutaitojaan

Arviointi:

Tehtävää arvioidaan formatiivisesti ja oppilaat voivat tehdä itsearvioinnin omasta työskentelystään tunnin päätteeksi. Jos ohjelmointia jatketaan useampi opetuskerta, voidaan itsearviointi tehdä jakson päätteeksi.

Ohjelmoinnin alkeet ScratchJr:lla, alkuopetus

29.9.2022

Ohjelmoinnin alkeet
tuntisuunnitelma alkuopetukseen

[redacted]

Aihepiirin valinta ja rajaus sekä aihepiirin oppimistavoitteet ja arviointi

- Näiden oppituntien tavoitteena on tutustuttaa pienet oppilaat **ohjelmointiin** ja **visuaaliseen ohjelmointiympäristöön** -ja kieleen, Scratchjunioriin.

- Ajatuksenani on käyttää myös ohjelmoinniin opetuksessa Varganemeny-menetelmästä tuttua abstraktion polkua, jossa asiat tehdään ensin konkreettisesti, sitten välineillä ja viimeisenä vaiheena kynäpaperitehtävinä tai tässä tapauksessa abstraktein muoto saavutetaan tietokoneella.

- **Arviointi** on jatkuvaa ja opettaja havannoi luokkaansa ja etenee opetuksessa opettajan mallista oppilaiden omiin kokeiluihin, kun näkee, että oppilaat ovat ymmärtäneet asian. Opettajan antaman palautteen lisäksi oppilaat saavat välittömän palautteen ohjelmoinnin onnistumisesta.

Työskentelyvälineet

- Scratchjunior komentopalat leikattuna ja lainoituna.

  https://www.scratchjr.org/pdfs/blocks.pdf

- Scratchjunior komentopalat leikattuna kullekin oppilasryhmälle

  https://www.scratchjr.org/assessments/block-labels.png

- Beebot-robotteja kullekin oppilasryhmälle
- Ipadit johon on asennettu Scratchjunior kullekin oppilasryhmälle

Oppikokonaisuuden voi jakaa kolmeen tuokioon tai yhdistää kaksi tai tuokiota samalle kerralle.

Ensimmäinen opetustuokio:

Opettaja on asetellut Scratchjuniorin tarvittavat komentopalat taululle. Joko leikattuna ja laminoituna valkotaululle tai sähköisenä versiona älytaululle. Mietitään yhdessä, mitä toimintaa kukin komentopala kuvaa. Sovitaan, että jokainen komento vastaa yhtä askelta. Tehdään yhdessä komentopaloista tointakäsky taululle ja opettaja toimii ensin robottina, joka kulkee toimintakäskyn määräämän radan. Tehtävää voidaan jatkaa yhdessä koko luokan kanssa niin, että halukkaat lapset pääsevät vuorotellen roboteiksi ja muu luokka tekee yhdessä komentoradan tai niin, että lapset tekevät pienissä ryhmissä toimintakäskyn monistetuilla ja leikatuilla komentopaloilla ja yksi ryhmästä toteuttaa sen.

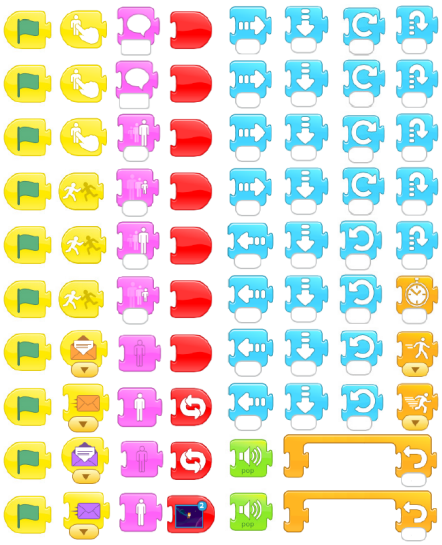
Toinen opetustuokio:

Aloitetaan oppitunti jälleen tutkimalla taululla olevaa Scratchjunior komentopaloilla rakennettua toimintakäskyä. Otetaan älytaululle tai dokumenttikameralle kuva Beebot-robotista ja pohditaan yhdessä, mistä komentopalojen komennot löytyvät Beebot-robotista. Ohjelmoidaan taululla oleva toimintakäsky Beebotilla. Oppilaat ohjelmoivat pienissä ryhmissä taululta olevan/olevat toimintakäskyt omalla robotillaan. Oppilaat suunnittelevat omia toimintakäskyjä ryhmissä omilla monistetuilla ja leikatuilla komentopaloilla ja toteuttavat ne.

Kolmas opetustuokio:

Aloitetaan jälleen oppituokio tutkimalla taululle Scratchjunior komentopaloilla rakennettua toimintakäskyä. Tällä kertaa ryhmät saavat Ipadit ja ohjelmointi tapahtuu oikeassa Scratchjunior-ympäristössä. Toteutetaan jälleen opettajan taululle luoma toimintakäsky. Tämän jälkeen riippuen oppilaiden kyvykkyydestä ja innostuksesta tutustutaan Scratchjuniorin ohjelmointiympäristöön laajemmalti.

Scratchjunior komentopalat:



Beebot-robotti

Kuva: OhjelmoinninABCVarhaiskasvatus_KangasVartiainen_2019.pdf

Ympäristötieto, ihminen ja ohjelmointi, 2.–3. lk

21.1.2021

Aihepiirin valinta ja rajaus

Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Ympäristötietoon, aihepiirinä "ihminen". Mielestäni ohjelmointia on helppo sisällyttää "ihminen" aihepiiriin, sillä siinä voidaan harjoitella koodauksen perusteita sekä ihmisen kehonosia ja tehtäviä helposti yhdessä.

Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Ihmisten kehonosat ja niiden toiminnot sekä ihmisen oma vaikutus kehon toimintoihin.

Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

Opitaan kehonosien nimet. Opitaan tärkeimmät elimet. Opitaan kehonosien ja elinten tehtävät. Opitaan ihmisen toiminnan vaikutus omaan kehoon.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

-Opitaan kehonosien nimet.

-Opitaan elinten nimet

- Opitaan kehonosien ja elinten tehtävät.

-Opitaan ihmisen toiminnan vaikutus omaan kehoon.

- opitaan koodauksen alkeita ja sanastoa

Miten opettaja tai joku muu arvioi opiskelijan osaamista? Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Ohjelmointia ja ympäristötiedon osa-aluetta arvioidaan erikseen. Itsearviointi ja vertaisarviointi tavoitteisiin nähden, sekä opettajan formatiivinen arviointi ja palaute.

Miten oppilas arvioi omaa osaamistaan?

Oppilas tietää ennen oppimiskokonaisuutta opettajan asettamat tavoitteet. Oppilas täyttää oppimiskokonaisuuden päätteeksi atsearvioinnin omasta osaamisesta ja kehittymisestä tavoitteisiin nähden.

Työskentelyvälineet ja opetusmenetelmät

Mitä välineitä oppikokonaisuudessa käytetään? Kynää ja paperia, omaa kehoa

Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Taululla voisi olla esillä koodauksen käsitteet ja avaukset, mitä ne tarkoittivat.

Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Tunnit sisältävät paljon konkretiaa ja hauskalla tavalla kehonosien ja tehtävien opettelemista.

Miten opetuksessa rohkaistaan yhteistyötä?

Oppilaat toimivat paljon pareittain/pienissä ryhmissä. Ennen tuntia käydään läpi yhteistyö merkitys ja toimintatavat.

Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Tunnit ovat suunniteltu niin, ettei juurikaan aiempaa ohjelmointi taustaa ole oppilailla. 2-3 luokille sopivat tuntisuunnitelmat.

2 x 45 min oppitunnit:

1. oppitunti

15 min johdatusta aiheeseen: tutkitaan parin kanssa/ ryhmissä ihmisen luurankoa ja nimetään osat jotka jo tiedetään.

15min kehonosien nimeämistä: Piirretään kartongille oma ihminen ja lisätään sinne tiedossa olevat kehonosat. Mennään toisen parin/ryhmän luo ja verrataan ihmisemme osia toisen parin ihmiseen. Täydennetään omia ihmisiä toisen ryhmän avustuksella.

15min koodaukseen johdattelu: toimitaan parin kanssa yhdessä. Pari antaa toiselle erilaisia käskyjä suullisesti liittyen toisen kehonosiin. Esim "kosketa oikealla kädelläsi vasempaa nillkkaa. Toista kolme kertaa. Kosketa polvellasi maata. Toista niin kauan kunnes saat uuden käskyn." Pohditaan tunnin lopuksi miltä tuntui antaa läskyjä ja miltä toisaalta tuntui vastaanottaa niitä.

2.oppitunti

Aikaisemmin on opittu tietämään ihmisen sisäelimistä maksa ja mahalaukku. Tällä tunnilla opettelemme niiden tehtävät samalla koodauksen alkeita opetellen. Aikaisemmilla tunneilla on myös käyty koodisanastoa läpi, kuten muuttuja, silmukka ja ehtolause.

25 min opetusvideo/teksti maksan ja mahalaukun tehtävistä

20min parin kanssa muodostetaan paperin ja kynän avulla koodi eli komentosarja maksan ja mahalaukun toiminnasta. Esim maksa:

jatkuu ikuisesti → säätele verensokeria → säätele aminohappopitoisuuksia→ valmista sappinestettä

Jos juo liikaa alkoholia → tulee rasvamaksaa → maksa tulehtuu

Muuten→

jatkuu ikuisesti → säätele verensokeria → säätele aminohappopitoisuuksia→ valmista sappinestettä

Lopuksi käydään yhdessä oppilaiden tekemiä koodeja läpi ja kerrataan, mitä koodisanastoa käskysarjoista löytyy.

Yhteistoiminnallista tutustumista algoritmikäsitteeseen, 3.lk

11.6.2020

YHTEISTOIMINNALLISTA TUTUSTUMISTA ALGORITMIKÄSITTEESEEN 3LK 4.5.2020

1. Oppitunti Liikuntatunti (YMPÄRISTÖ: jumppasalissa tai ulkona)

YHTEISTOIMINNALLISTA JA LIIKUNNALLISTA TUTUSTUMISTA ALGORITMI-KÄSITTEESEEN

Seuraavia leikkejä voi käyttää, minkä tahansa oppiaineen oppitunnin alussa, lomassa tai lopussa. Tavoitteena on yhteistoiminnallisesti ja liikunnallisesti ymmärtää algoritmin käsite= vaiheittain etenevä ohje. (Tietokonekielessä algoritmi on vaiheittain etenevä ohje ongelman ratkaisemiseksi.) Tehtävät voi tehdä pareittain tai pienissä ryhmissä. Opettaja näyttää aina ensin mallisuorituksen kaikkien oppilaiden tehdessä tehtävää opettajan ohjauksessa. Parit ja ryhmät pyritään mielellään arpomaan, jos se ryhmän koostumuksen huomioiden on vain mahdollista. Opettaja voi myös määrätä parit.

PELI/LEIKKI	TAVOITTEET	AIKA	VÄLINEET	YHTEISTYÖMUOTO	ARVIOINTI
Motivointileikki **Kapteeni käskee**- Jos ohjaaja sanoo Kapteeni käskee juokse paikallasi, tulee leikkijän juosta paikallaan, mutta, jos ohjaaja sanoo pelkästään juokse paikallasi ja pelaaja ryhtyy juoksemaan paikallaan, putoaa hän leikistä pois. Ensin ope mallittaa ja sitten pienemmissä ryhmissä.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely ryhmässä muut huomioiden	10 min	Ei välineitä	Koko ryhmä, 3-4 oppilaan ryhmät	Opettaja arvioi seuraamalla eri ryhmiä. Peukutetaan omaa suoritusta: Oman ohjauksen sujuminen ja oma ohjeiden seuraaminen ja kuunteleminen.
Robottileikki 1-Parisi on robotti ja sinun pitää antaa robotille tarkat ohjeet, miten robotti esim. potkaisee palloa, tekee vatsalihas liikkeen tai heittää korin vaihe vaiheelta. Ensin ope mallittaa ja sen jälkeen jakaannutaan pareihin open valitsemalla tavalla. Ensin ope mallittaa ja sitten tehdään pienemmissä ryhmissä.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa	10 min	Ei välineitä	Koko ryhmä, parityöskentely	Opettaja arvioi seuraamalla. Peukutetaan omaa suoritusta: Oman ohjauksen sujuminen ja oma ohjeiden seuraaminen ja kuunteleminen.
Robottileikki 2-toinen parista on robotti ja toinen ohjaaja. Ohjaaja miettii ensin, minne haluaa robotin kulkevan. Sitten ohjaaja sanoo käskyt ensimmäisellä kerralla sanallisesti ja toisella kerralla koodiavaimen numerolla. Koodiavain oppilaalle 1=Mene eteenpäin, 2=Pysähdy, 3= Käänny paikallasi oikealle, 4=Käänny paikallasi vasemmalle. Ensin ope mallittaa ja sitten pienemmissä ryhmissä.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa - oikean ja vasemman vahvistaminen	10 min	Koodiavain-lappu jokaisella 1=Mene eteenpäin 2=Pysähdy 3= Käänny paikallasi oikealle 4=Käänny paikallasi vasemmalle	Koko ryhmä, parityöskentely	Opettaja arvioi seuraamalla. Peukutetaan omaa suoritusta: Oman ohjauksen sujuminen ja oma ohjeiden seuraaminen ja kuunteleminen.
Ruutusuunnistus- piirtäkää 5x5 askelta oleva ruudukko hiekkakentälle tai hyödyntäkää koulun pihan ruudukoita. Merkitkää yläruutujen yläpuolelle numeroita/ kirjaimia/ esineitä. Pyydä pariasi etenemään ruudukossa ylös/alas/oikealle/vasemmalle. Ohjaa hänet jollekin numerolle/esineelle /kirjaimelle	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa - ylös, alas, oikean ja vasemman vahvistaminen	15 min	Pieniä esineitä luokasta tai merkkikartioita/nauhoja/palloja jne. liikuntavarastosta	Parityöskentely	Opettaja arvioi seuraamalla. Keskustellaan lopuksi yhdessä, onko helppo kuunnella, noudattaa tai antaa ohjeita. Mikä on helpointa ja mikä vaikeinta?

2. Oppitunti Kuvaamataidon/matematiikan oppitunti

YHTEISTOIMINNALLISIA KÄDENTAITOJA KEHITTÄVIÄ TEHTÄVIÄ (YMPÄRISTÖ: Luokkatila)

Tavoitteena on yhteistoiminnallisesti kädentaitoja samalla harjoittaen ymmärtää algoritmin käsite. Tehtävät voi tehdä pareittain tai pienissä ryhmissä. Opettaja näyttää aina ensin mallisuorituksen kaikkien oppilaiden tehdessä tehtävää opettajan ohjauksessa. Parit ja ryhmät pyritään mielellään arpomaan, jos se ryhmän koostumuksen huomioiden vain on mahdollista. Opettaja voi myös määrätä parit.

PELI/LEIKKI	TAVOITTEET	AIKA	VÄLINEET	YHTEISTYÖMUOTO	ARVIOINTI
Motivointi: Mitä geometristä kuviota ajattelen? Vastataan kysymykseen kyllä tai ei. **Robottileikki 3** -Toinen parista tai yksi ryhmän jäsenistä saa tehtäväkseen ohjata saamansa piirrostehtävän, joka on ruutupaperilla (helppoja: neliö, suorakulmio, talo). Ohjeina x ruutua ylös/alas/oikealle/vasemmalle. Jokainen on vuorollaan ohjaaja.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa -- ylös, alas, oikean ja vasemman vahvistaminen	15 min	Opettajan valmiit mallit. Jokaiselle ryhmän jäsenelle annetaan oma tehtävä ohjattavaksi muille. Lyijykynä, pyyhekumi, numerokortit 1-5	Parityöskentely tai 2-3 oppilasta ryhmässä	Opettaja arvioi , millaisia tuotoksia oppilaat saavat aikaan ja miten pari-/ryhmätyöskentely sujuu. Oppilaat suorittavat itsearviointia näyttämällä numeron (ei pidä paikkansa)1-5(pitää paikkansa) 1.Ohjeiden seuraaminen oli minulle helppoa.
Robottileikki 4 -Vaikeampi versio on ensin suunnitella oma piirros ja ohjata se muille.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa -- ylös, alas, oikean ja vasemman vahvistaminen	15 min	Ruutupaperi, jolle jokainen suunnittelee oman melko yksinkertaisen mallin. Kuvion piiri <16 ruutua.	Parityöskentely tai 2-3 oppilasta ryhmässä	2. Ohjeiden antaminen oli minulle helppoa. 3. Piirroksestani tuli samanlainen kuin mitä alkuperäinen oli. 4. Ryhmässä työskentely sujui minulta hyvin.
Robottileikki 5 Rakennetaan valituilla välineillä pöydälle kuvio, jota ei näytetä parille tai muille ryhmän jäsenille. Ohjataan pari/muut oppilaat rakentamaan samanlainen kuvio omalle pöydälle. Verrataan kuvioita toisiinsa.	-ohjeiden kuunteleminen -tarkkojen ohjeiden antaminen -sujuva ja joustava työskentely parin kanssa -- ylös, alas, oikean ja vasemman, alapuolelle, yläpuolelle	15 min	Murtokakut, tangram-palat, legot, multilink-palikat tms.	Parityöskentely tai 2-3 oppilasta ryhmässä	Opettaja arvioi, millaisia tuotoksia oppilaat saavat aikaan ja miten pari-/ryhmätyöskentely sujuu. Oppilaat arvioivat samat väitteet, mutta asetutaan viivalle luokassa, jossa toisessa päässä on en onnistunut ja toisessa päässä onnistui hyvin.

Projekti jatkuu siten, että seuraavalla kerralla oppilaat kertovat ensin suullisesti erilaisten **arkipäivän toimintojen ohjeet**. Esim. hampaiden pesu, voileivän tekeminen, nukkumaan meneminen jne. **Suullisen harjoituksen jälkeen** nämä **kirjoitetaan** vihkoon. **Opetellaan algoritmi-sana** ja sen sisältö. Tämän jälkeen voidaan harjoitella kirjoittamaan niitä Scratchin visuaalisella ohjelmointikielellä. Opitaan sanat tapahtuma, muuttuja ja silmukka. Tehdään tanssiesitys, joka myös **kirjataan visuaaliseen ohjelmointikieleen** ja otetaan toistorakenteita käyttöön. Lopulta Scratchin visuaalinen ohjelmointikieli tulee tutuksi ja voidaan aloittaa **helppojen harjoitusten tekeminen Scratchillä**.
Tutkitaan myös **ruokareseptejä**.

Ohjelmoinnin harjoituksia ilman tietokonetta, 3.lk

Kristiina Länsiö, CC BY-SA 4.0

10.4.2021

Minkä tehtävän valitsit ja miksi?

Ensimmäinen tunti

Ruutupaperi tai vihko

Opettaja pyytää laittamaan aloituspisteen vihkoon keskelle sivua tai esim. lasketaan vasemmasta reunasta ruutuja 10 ja ylhäältä ruutuja 15. Piirretään taululle/vihkoon kompassiruusu ja kerrataan ilmansuunnat.

Annetaan toimintaohjeet. Piirrä viiva nykyisestä pisteestä seuraavasti:

Mene 5 ruutua etelään/alas

Mene 2 ruutua länteen/vasemmalle

Mene 1 ruutu etelään/alas jne

Muodostuu kuva:

Yhteisen piirtämisen jälkeen oppilaat suunnittelevat toisilleen saman tyyppisen tehtävän. Oppilas piirtävät ensin pienen kuvion ja miettivät kuinka piirrosohjeet annetaan nuolien avulla. Nuolet piirretään vihkoon ja tehtävä annetaan oppilaskaverille ratkaistavaksi.

Toinen tunti

Arvaa, mitä lukua ajattelen?

Opettaja tai oppilas keksii luvun joltain lukuväliltä, esimerkiksi 0-1000. Leikin vetäjä saa vastata kysymyksiin joko ”Kyllä” tai ”Ei”. Oppilaiden tavoite on arvata luku mahdollisimman vähällä määrällä kysymyksiä.

Ohjeet: Opettaja aloittaa leikin piirtämällä taululle lukusuoran, jonka toisessa päässä on 0 tai toisessa päässä esimerkiksi 1000. Opettaja keksii luvun ja alkaa arvuutella luokalta, mikä luku voisi olla kyseessä. Opettaja saa vastata kysymyksiin ”Kyllä” tai ”Ei” ja voi kirjoittaa taululle ylös jo

arvatut luvut. Kannattaa miettiä, millainen on hyvä kysymys, jolla saa rajattua mahdollisimman monta väärää lukua pois.

Koodari (jos-niin -ehtolause)
Yksi leikkijöistä on koodari. Hän kertoo ohjeita seuraavasti:
JOS minä taputan, NIIN te taputatte.
JOS minä vilkutan, NIIN te vilkutatte.
Vaihdetaan sääntöjä siten että oppilaat tekevät eri asiaa kuin opettaja, esim.
JOS minä hymyilen, NIIN te irvistätte.
JOS minä hyppään, NIIN te menette kyykkyyn. JOS menen kyykkyyn, NIIN te hyppäätte. JOS minä taputan, NIIN te tömistätte. JOS minä taputan, NIIN te vilkutatte.
Koodari tekee liikkeitä ja muut leikkijät yrittävät seurata niitä ohjeen mukaisesti. Selkeästä virheestä putoaa pelistä. Opettaja seuraa luokkaa ja kertoo ketkä putoavat pelistä ja jäävät istumaan paikalleen. Viimeinen leikkijä pääsee seuraavaksi koodariksi.
Muunnoksia: Voidaan lisätä ehtoihin JA- ja TAI-käskyt.
Esimerkiksi JOS minä taputan JA hymyilen, NIIN te hyppäätte, tai JOS minä hyppään TAI tömistän, NIIN te vilkutatte.

Tämän jälkeen tutustuimme kolmasluokkalaisten kanssa Code.org
sivuston koodaustunti tehtäviin. Siinä harjoitellaan edellisen tehtävän mukaisia ohjeiden antamisia.

Mitä opetussuunnitelman tavoitteita tehtävä tuki?
T4 kannustaa oppilasta esittämään päättelyään ja ratkaisujaan muille konkreettisin välinein, piirroksin, suullisesti ja kirjallisesti myös tieto- ja viestintäteknologiaa hyödyntäen
T5 ohjata ja tukea oppilasta ongelmanratkaisutaitojen kehittämisessä
T14 innostaa oppilasta laatimaan toimintaohjeita tietokoneohjelmina graafisessa ohjelmointiympäristössä

Minkälaista tietoa sait oppilaiden osaamisesta suhteessa niihin ops:in tavoitteisiin, joita tehtävä mielestäsi tuki?
Oppilailla oli suuri innostus tunnin aiheeseen. Parityönä code.org sivuston ohjelmointiharjoitukset sujuivat hienosti. Toiselle selittäminen kehittää matikkapuhetta, vuorovaikutustaitoa ja antaa myös ymmärrystä asiaan,

Lähteet: Ville_alakoulun_ohjelmointiopas.pdf
https://pulmaario.luma.fi/matematiikka/
Kuva: Pinterest

Liikuntaa ja ohjelmointia, 3.lk

15.3.2021

Tuntisuunnitelma

Aihepiirin valinta ja rajaus

Tämä kokonaisuus on suunniteltu kolmosluokkalaisille, mutta soveltamalla suunnitelmaa voisi käyttää nuoremmilla tai vanhemmilla oppilailla. Näillä oppitunneilla yhdistetään ohjelmointia liikuntaan ja mahdollisesti ympäristöoppiin.

Olettamuksena on, että oppilaat ymmärtävät ohjelmoinnista jo jotain perusperiaatteita ja näiden oppituntien aikana lähdetään tutustumaan komentoihin konkretian kautta. Olisi hyvä, että ohjelmoinnista olisi siis jo keskusteltu ja esimerkkinä oltaisiin vaikka käytetty tehtävää, jossa oppilaat antavat komentoja ja opettaja liikkuu niiden mukaisesti.

Tämän suunnitelman taustalla on tavoite toisenlaisesta lähestymistavasta perinteiseen kuntopiiriin tai voimistelutuntiin, sillä yleensä nämä tunnit ovat hyvin opettajajohtoisia. Ohjelmoinnin yhdistäminen liikuntaan sopii voimisteluun hyvin. Ympäristöopin yhdistäminen liikuntaan tulee puolestaan hyvin luontevasti, sillä eläinten eri liikkumistapojen matkiminen on oppilaille tuttua jo varmasti varhaiskasvatuksesta lähtien. Tämän tunnin voi hyödyntää tutustumisena voimisteluun/kuntopiiriin tai se voi olla tämäntyyppisen voimistelujakson kertaus. Tästä riippuen opettaja voi muokata tehtävänantoa. Tärkeää on kuitenkin huolehtia opitun kokoamisesta, jotta oppitunneista saadaan kaikki irti eivätkä yksittäiset osa-alueet jää kokonaan irrallisiksi.

- Jos kyseessä on kertaus, liikkeiden oletetaan olevan tuttuja ja tällöin opettaja voi lisätä tehtävänantoon vaatimuksen tiettyjen liikkeiden sisällyttämisestä toteutukseen. (Ks. Taulukko 3.)

- Jos taas kyseessä on tutustuminen voimisteluun, oppilaille on hyvä antaa vapaampi tehtävänanto ja ajatuksena on, että tästä aloitetaan liikunnassa voimistelujakso käymällä myöhemmin läpi oppilaiden valitsemia liikkeitä ja opettamalla uusia. Tehtävään voidaan tällöin ottaa myös viittaus ympäristöoppiin. Kolmannella luokalla tutustutaan monipuolisesti Suomen eläinkuntaan. Tehtävää varten kannattaa valita eläimet/linnut/kalat/hyönteiset, joihin oppilaat ovat jo tutustuneet ympäristöopissa. (Ks. Taulukko 4.)

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Tavoitteet käydään yhteisesti läpi tehtävänannon yhteydessä. Opettaja liikkuu oppituntien aikana oppilaiden parissa ja merkitsee itselleen taulukkoon muistiinpanoja oppilaiden työskentelystä. Lopuksi oppilaat täyttävät vielä itsearviointilomakkeen. Alla olevissa taulukoissa olen kuvannut esimerkkitavoitteita oppitunnille. Tavoitteet on valittu Opetussuunnitelman perusteiden mukaisesti. Näitä voi muokata sen perusteella, mitä oppitunnilla haluaa painottaa.

	Ohjelmointi: Ymmärtää ohjeiden tarkkuuden merkityksen lopputulokselle	**Liikunta:** Suunnittelee tehtävänannon mukaisen liikesarjan	**Liikunta:** Harjoittelee yhteistyötä	**Liikunta:** Osoittaa aktiivisuutta ja halua saattaa tehtävä valmiiksi	(**Ympäristöoppi:** Tunnistaa eläimen ja sille ominaisen tavan liikkua ja osaa yhdistää sen mielekkäästi tehtävänantoon)
Pari1					
Pari2					
Pari3					
Pari4					

Taulukko 1. Opettajan arviointilomake

	Erinomaisesti	Hyvin	Melko hyvin	Harjoittelen vielä
Keskityin tehtävään				
Yritin parhaani				
Olin aktiivinen				
Kuuntelin pariani				
Autoin pariani				
Sain liikesarjan suunniteltua				
Liikesarjassamme oli kolme liikettä				
Liikesarjassa käytimme meille arvotun eläimen tapaa liikkua				

Oman liikesarjan pystyi toteuttamaan ohjeillamme				
Toisen parin liikesarjan pystyi toteuttamaan heidän ohjeillaan				

Taulukko 2. Itsearviointilomake

<u>Työskentelyvälineet ja opetusmenetelmät</u>

Oppitunti 1.

Opettaja esittelee tehtävän ja kertoo tehtävänannon. Tehtävänanto kannattaa kirjoittaa taululle tai tulostaa pareille mukaan otettavaksi. (Huom! Muokkaa tehtävänantoa sopimaan koululta löytyviin välineisiin.)

Käykää läpi termit: voimistelu ja liikesarja sekä opettajan odotukset niitä kohtaan. Palauttakaa mieleen ohjelmoinnin perusperiaatteet, jotta oppilaat ymmärtävät ohjeen merkityksen. Muista myös selittää, mitä tarkoitat eläimen yhdistämisellä tehtävään. Tai vaihtoehtoisesti palauta oppilaiden mieleen voimistelu/kuntopiiriliikkeet.

<u>Esimerkki:</u> Oppilaat saavat eläimekseen karhun. He suunnittelevat liikesarjan, jossa he kävelevät kolme askelta karhukävelyä, pyörähtävät kontallaan ympäri ja tekevät kuperkeikan ja jatkavat kolme askelta karhukävelyä.

Tehtävänanto:

1. **Tehtävänänne on tehdä voimistelu- tai kuntopiirisarja. Sarjassa tulee olla vähintään KOLME liikettä.**
2. **Käytä sarjassa teille arvottuja voimistelu/kuntopiiriliikkeitä. <u>TAI</u> Ota sarjassa huomioon teille arvottu eläin. Miten tämä eläin liikkuu?**
3. **Kirjoittakaa ohje sarjaanne tarkasti paperille. Muista, että jokainen vaihe on tärkeä, jotta ohjettanne voidaan käyttää oikein.**
4. **Palauta paperi lopuksi opettajalle.**

(5. Kuvatkaa oma liikesarjanne videolle oman ohjeenne mukaisesti ja palauttakaa video Google Classroomiin.)

Oppilaat jaetaan pareihin. Kukin pari nostaa hatusta jonkin voimisteluliikkeen TAI eläimen nimen/kuvan. Allaolevassa taulukossa on esimerkkejä tehtävään sopivista liikkeistä ja eläimistä. Lähteinä voimistelu- ja kuntopiiriliikkeisiin on käytetty Suomen voimisteluliiton sekä Valmennuspankin internetsivuja.

Kuperkeikka	Kärrynpyörä	Krokotiilikävely	Sivukuperkeikka
Varsahyppy	Kyykkykävely	Yhdenjalanhyppy	Yhdenjalanseisonta
Tramppa-hyppy	Karhukävely	Piruetti	Taaksepäinkuperkeikka
Vatsarutistus	Lantionnosto	Punnerrus	Lankku
Kylkipito	Burpee	Tuulimylly	Vatsalihaskierto

Taulukko 3. Voimistelu/kuntopiiriliikkeet

TAI

Karhu	Kurki	Hylje	Myyrä
Sisilisko	Sorsa	Orava	Hauki
Heinäsirkka	Lokki	Jänis	Perhonen

Taulukko 4. Eläimet

Oppilaille jaetaan tarvittavat välineet. Opettaja voi valita saavatko oppilaat käyttää tehtävässään liikuntavälineitä ja miten välineet jaetaan parien kesken. Jos aikaa on niukasti, voi tehtävä olla helpompi ilman välineitä.

Osoita oppilaille työskentelyyn sopivat tilat. Vaikka kyseessä on liikuntatunti, tehtävään ei välttämättä tarvita liikuntasalia. Oppilaita voi jakaa muihinkin vapaisiin tiloihin.

Oppilaat työskentelevät. Kierrä ja ohjaa oppilaita. Auta eteenpäin, mutta älä ohjaa liikaa.

Muistuta oppilaita palauttamaan välineet ja ohjepaperit opettajalle.

Oppitunti 2.

Aloita oppitunti palauttamalla oppilaiden mieleen edellisen tunnin työskentely ja tehtävänanto. Muistuta nyt erityisen tarkkaan ohjeiden kirjaimellisesta noudattamisesta. Käykää läpi tämän tunnin tehtävänanto. Jaa oppilaille eri parin ohje heidän suunnittelemastaan liikesarjasta. Jos oppilaat ovat nopeita, voit antaa heille useamman parin ohjeet tunnin aikana.

Tehtävänanto:

1. **Lukekaa teille annetut ohjeet tarkasti.**
2. **Tehkää liikesarja ohjeiden mukaisesti. Muistakaa tehdä vain se, mitä paperissa lukee eikä mitään muuta!**

(3. Kuvatkaa liikesarja ja palauttakaa video Google Classroomiin.)

Tehtävässä ei ole pakko käyttää videointia lainkaan. Näin oppitunti lyhenee ja yksinkertaistuu. ja mahtuu paremmin kahteen oppituntiin. Videointi kuitenkin varmasti herättää oppilaiden mielenkiintoa tehtävää kohtaan. Samalla työskentelyyn tulee myös luontaisesti eri rooleja, jotka vaativat oppilailta työnjakoa. Jos annat oppilaille ohjeeksi kuvata liikesarja, huolehdi, että kaikilla pareilla on jokin laite, jolla kuvaaminen onnistuu.

Työskentelyn jälkeen oppilaat saavat katsottavakseen toisten parien tekemät videot omien ohjeidensa perusteella. Oppilaat vertaavat omaa ohjettaan ja omaa videotaan toisen parin suoritukseen videolla.

Lopuksi käydään luokkakeskustelu havainnoista. On tärkeää vielä yhteisen keskustelun ja esimerkkien kautta saada oppilaat kiinnittämään huomiota ohjeiden tarkkuuden merkitykseen ohjelmoinnissa. Miten toinen pari tulkitsi ohjeet? Olivatko ohjeet onnistuneet? Jos olivat/eivät, niin miksi?

Samalla voidaan keskustella siitä, missä muissa tilanteissa ja yhteyksissä tarkoilla ohjeilla on iso merkitys. Oppilaita voidaan kannustaa pohtimaan erilaisia arkisia laitteita tai tilanteita, joissa on taustalla algoritmista ajattelua. Tässä keskustelussa olisi viimeistään oppilailla tilaisuus osoittaa aikaisempaa tietämystään ohjelmoinnista.

Lopuksi oppilaat täyttävät vielä itsearviointilomakkeet.

Tämän tehtävän jälkeen olisi mielekästä siirtyä ohjelmointiin tietokoneilla esimerkiksi pelin ohjelmointiin Scratchillä tai kokeilemaan Code tai Khan Academy - sivustojen mahdollisuuksia.

Lähteet:

Code
code.org
Khan Academy
https://www.khanacademy.org/coach/welcome
Perusopetuksen opetussuunnitelman perusteet
https://www.oph.fi/sites/default/files/documents/perusopetuksen_opetussuunnitelman_perusteet_2014.pdf
Scratch
https://scratch.mit.edu
Suomen Voimisteluliitto
https://www.voimistelu.fi/fi/Silta/Materiaalipankki/Lajitaito/Joukkuevoimistelu
Valmennuspankki
https://valmennuspankki.wordpress.com/liikepankki/

Ohjelmointi-suunnistus Seppo-pelialustalla, 3. lk

Mira Christiansen, CC BY-SA 4.0

2.5.2022

Ohjelmointi-suunnistus Seppo-pelialustalla, 3. luokka

Aihepiirin valinta ja rajaus

Ohjelmointi-suunnistus yhdistää ohjelmoinnin ja liikunnan. Tavoitteena on tutustua koodaukseen / koodinpurkuun sekä ohjelmoinnin alkeisiin toiminnallisten ja vuorovaikutteisten tehtävien avulla, toimia yhteistyössä parin kanssa sekä liikunnan ilo. Suunnistus toteutetaan oman koulun pihalla tai muussa soveltuvassa lähiympäristössä. Suunnistuksen voi toteuttaa myös sisäsuunnistuksena koulurakennuksessa.

Tietoa Seposta: Pedagogiikka — Seppo

Ohjelmointi-suunnistukseen käytettävä aika on 2 oppituntia eli kaksoistunti. Opetus alkaa luokassa, jossa käydään läpi ohjeet ja tutustutaan sähköiseen pelialustaan. Alustalle on merkitty toimintapisteet pienten vihjeiden avulla, kuten ”täältä kuuluu välituntisin riemunkiljahduksia ja vauhti on päätä huimaava” (keinut) tai ”hiekkaa kengässä” (hiekkalaatikko). Tehtäväpaikoilla suoritetaan parin kanssa annettu tehtävä, joka liittyy jollain tapaa ohjelmoinnilliseen ajatteluun. Tehtävien vastaukset tallennetaan Seppo-alustalle, johon vastauksen voi lähettää kirjoitettuna, kuvana, ääni- tai videotallenteena. Opettajan rooli on toimia pelin ohjaajana, kannustaa ja antaa palautetta suorituksista. Oppitunti päätetään omaan luokkaan, jossa käydään yhteinen lyhyt palaute- / arviointikeskustelu.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Arviointi kohdistuu sekä ohjelmoinnin osaamiseen että liikuntatunnin työskentelyyn. Opettaja kannustaa ja antaa oppilaille palautetta tunnin edetessä Seppo-sovelluksen kautta ohjaajan roolissa. Lopuksi käydään yhteinen arviointikeskustelu luokan kanssa. Parit saavat kertoa onnistumisistaan, haasteista ja muista huomioistaan. Pohditaan, mitä uutta opittiin. Jokainen antaa lyhyen arvion (usein käytän ns. peukkuarviointia eli esitän kysymyksiä, joihin oppilaat vastaavat näyttämällä peukkua ylös / alas / sivulle) omasta työskentelystään. Opettaja antaa tässä kohtaa vielä yhteistä palautetta tunnista.

Ohjelmoinnin tavoitteet, 3. luokka, matematiikka (ops.edu.hel.fi):

S1 Ajattelun taidot

• Kehitetään oppilaan taitoja löytää yhtäläisyyksiä, eroja ja säännönmukaisuuksia.

• Syvennetään taitoa vertailla, luokitella ja asettaa järjestykseen, etsiä vaihtoehtoja systemaattisesti, havaita syy- ja seuraussuhteita sekä yhteyksiä matematiikassa.

• Tutustutaan graafiseen ohjelmointiympäristöön.

”Konkreettia ja toiminnallisuus ovat keskeinen osa matematiikan opetusta ja opiskelua. Oppimista tuetaan hyödyntämällä tieto- ja viestintäteknologiaa.”

”Oppilailla on mahdollisuus vaikuttaa työtapojen valintaan. Oppilaat työskentelevät sekä itsenäisesti että ryhmässä. Oppimispelit ja -leikit ovat tärkeitä ja oppilaita motivoivia työtapoja.”

Liikunnan tavoitteet, 3. luokka (ops.edu.hel.fi):

S2 Sosiaalinen toimintakyky

Opetukseen valitaan myönteistä yhteisöllisyyttä lisääviä pari- ja ryhmätehtäviä, leikkejä, harjoituksia ja pelejä, joissa opitaan ottamaan toiset huomioon ja auttamaan muita sekä tehtäviä, joissa opitaan vastuun ottamista omasta toiminnasta, yhteisistä asioista ja säännöistä.

S3 Psyykkinen toimintakyky

Opetuksessa käytetään tehtäviä, joissa opitaan pitkäjännitteisesti ponnistelemaan yksin ja yhdessä muiden kanssa tavoitteen saavuttamiseksi. Yhteisillä tehtävillä harjoitellaan vastuun ottamista. Iloa ja virkistystä tuottavilla liikuntatehtävillä autetaan myönteisten tunteiden kokemista, jotka vahvistavat pätevyyden kokemuksia ja myönteistä minäkäsitystä.

”Ilmiöpohjaisessa opiskelussa tavoitteet ja sisällöt johdetaan laaja-alaisen osaamisen ja eri oppiaineiden tavoitteista. Liikunnan tavoitteita ja sisältöjä integroidaan eri oppiaineiden kanssa ilmiöoppimisen kokonaisuuksiksi. Työskentelyote on tutkiva ja siinä käytetään mahdollisuuksien mukaan liikunnan asiantuntijoita sekä menetelmiä ja lähteitä. Pelien ja leikkien avulla voidaan toteuttaa ilmiöoppimisen tavoitetta joustavista opetusjärjestelyistä...”

Työskentelyvälineet ja opetusmenetelmät

Tunnilla tarvittavat välineet ovat hyvät liikuntavarusteet ja tabletti tai puhelin, johon on ladattu Seppo-sovellus. Oppilaat tarvitsevat myös muistiinpanovälineet koodien purkamista varten.

Käytettävään sovellukseen tutustutaan ensin luokassa, jotta oppilaat osaavat käyttää sitä sujuvasti. Tarvittaessa opettaja auttaa teknisissä pulmissa tai mikäli tehtävän ymmärtämisessä on haasteita.

Tehtävät voivat olla esimerkiksi seuraavanlaisia:

1. Atbash-salauksella kirjoitetun viestin purku: toimintapisteellä on jokin salakirjoitusviesti, joka oppilaiden tulee ratkaista.

”Atbash-salakirjoitus on myös hyvin vanha salakirjoitusmenetelmä, jossa jokainen kirjain korvataan aakkosten "vastakkaisella" kirjaimella. Atblash-salauksella salakirjoitettu sana KISSA olisi siis kirjainyhdistelmä SUKKÖ” (Ohje Summamutikasta)

Purettu koodi eli salakirjoitusviesti lähetetään Seppo-alustalle ja peli antaa vastauksen, onko ratkaisu oikea.

2. Kuviokoodit: Tehtävään on merkitty kuviokoodit ja kirjoitettu, mitä liikettä mikäkin kuvio tarkoittaa. Oppilaiden tulee liikkua koodin osoittamaan paikkaan ja tämän jälkeen valita pelialustalla monivalintatehtävässä, mitä he näkevät edessään (esim. ison kiven, penkin, puun tms.). Jos he ovat onnistuneet liikkumaan koodin mukaisesti oikein, vastaus menee oikein.

3. Salaviesti numeroilla: Muutetaan aakkoset numeroiksi: 1=A, 2=B… Oppilaiden tulee muuttaa oma etunimensä numeroiden avulla salaviestiksi ja lähettää äänitallenne pelialustalle, jossa lausutaan oma nimi. Esimerkiksi Maija: 13-1-9-10-1. Opettaja tarkistaa vastaukset ja antaa palautetta tehtävästä. (Ohje numerosalaviestiin Summamutikasta)

4. Ruudukko-salakirjoitus: Oppilaat ratkaisevat salakirjoitetun viestin ja kirjoittavat oikean vastauksen pelialustalle. Lisäksi tehtävänä on kirjoittaa kepillä maahan (tai sisällä esim. paperille tai taululle) pariaan kuvaava adjektiivi ruudukkosalakirjoitusta käyttäen ja lähettää tästä kuva pelialustalle. (Ohje ruudukkosalakirjoitukseen Summamutikasta)

5. Radan kulkeminen silmät kiinni: Tehtävänä on kuljettaa pari, jolla silmät kiinni, tietyn radan läpi. Rata on merkitty pihalle / esim. sisällä kartioin. Pari antaa toiselle ohjeita ”eteen”, ”taakse”, ”oikealle”, ”vasemmalle” -käskyin. Suorituksen voi videoida, jos se onnistuu, tai kirjoittaa pelialustalle vastauskenttään, miten tehtävä sujui ja pääsikö pari radan loppuun. (Idea tehtävään: Opentunti - Suunnitelma - Liikunnallinen ohjelmointi)

Tässä muutamia ehdotuksia toimintapisteisiin. Toimintapisteitä kannattaa olla reilusti, jotta niihin ei tule ruuhkaa ja tekemistä riittää koko (kaksois-)tunniksi. Yksi mahdollisuus on antaa parien ensin suunnitella toimintapiste, jolloin tehtävä osallistaa oppilaat entistä paremmin. Tällöin parit jättäisivät oman toimintapisteensä välistä.

Oppilaita motivoi liikunta, tehtävien lomassa saatu kannustava palaute ja yhteistyö parin kanssa hauskojen ja sopivasti haastavien tehtävien parissa. Pelillisyys itsessään aktivoi oppilaita, sillä tehtäviä suorittamalla pääsee eteenpäin. Tehtävissä pääsee käyttämään luovuuttaan. Lisäksi motivointina voi olla jokin pieni palkkio hyvästä suoriutumisesta ja yrittämisestä.

Ohjelmointia ruudukossa ja kertolaskujen kertaus, 3. lk

5.5.2022

3. luokka ohjelmointi ruudukossa ja kertolaskujen kertaus

Suunnittelin 3. luokan pienryhmälleni (8 oppilasta) ohjelmointitunnin kertolaskujen kertaamiseen. Alkuvuodesta olemme tehneet kirjan ohjelmointitehtäviä, joissa leppäkerttu täytyy ohjata kukan luo sekä noudattamalla ohjeita että luomalla ohjeet itse. Tehtävätyyppi on oppilaille siis tuttu. Edellisellä oppitunnilla palautetaan tehtävätyyppi kuitenkin pikaisesti mieleen.

Olen suunnitellut yhden oppitunnin mittaisen tehtävän, mutta oman ryhmäni tuntien tässä tehtävässä saatetaan käyttää myös osa seuraavan tunnin äidinkielen tunnista. Tunnilla työskennellään parin kanssa. Parityöskentelyssä oppilailla on vielä haasteita, he tarvitsevat paljon ohjausta ja muistutusta, että keskittyvät yhdessä annettuun tehtävään. Eriyttäminen tapahtuu parivalinnoilla sekä valituilla kertotaulutehtävillä (osalle pareista omat tehtävät). Arviointi toteutetaan seuraamalla oppilaiden tuntityöskentelyä sekä tunnin lopussa itsearvioinnilla.

Välineet:

- maalarinteippi
- kuvakortit (tai muut vastaavat ruudukkoon)
- leimakortti (liitteenä)
- ohjelmointikortit x4 per pari (pohja liitteenä)
- kertotaulutehtävät (liitteenä)
- itsearviointilomake (liitteenä)

Tunnin alkuun ope kertoo ja kirjaa taululle näkyviin tunnin tavoitteet.

- parin kanssa työskentely keskittyen annettuun tehtävään
- vaiheittaisten ohjeiden anto sekä niiden noudattaminen
- kertotaulujen kertaus

Tehtävän kulku.

Oppilaiden tavoitteena on kerätä tunnin ajalta erilliseen korttiin 4 leimaa. Leiman saa, kun ensin on ohjeistettu kaveri ruudukossa oikeaan kohtaan ja siihen liittyvä pieni tehtävämoniste on tehty yhdessä. Molemmat oppilaat pääsevät siis ohjeistamaan sekä noudattamaan ohjeita kaksi kertaa.

Ohjelmointiosuus: Lattialle on etukäteen tehty maalarinteipillä 4x4 ruudukko. Tarvittaessa ruudukoita voi tehdä myös useamman. Ope on laittanut ruutuihin valmiiksi kuvakortteja tai muita vastaavia symboleita. Oppilailla on yhteensä 4 erilaista ohjelmointikorttia. Vain toinen parista näkee yhden kortin ja sen avulla ohjeistaa toisen liikkumaan ruudukossa oikean kuvakortin luo. Ohjelmointikortissa on 4x4 ruudukko, johon ope on piirtänyt valmiiksi reitin, jota noudattamalla pääsee oikean kuvakortin luo. Ohjeistavan oppilaan on siis sanallistettava open piirtämä reitti toiselle oppilaalle. “Liiku eteenpäin 1, käänny vasemmalle.” Kun päästään kuvakortin luo, kerrotaan kuva opelle ja saadaan sitä vastaava tehtävä.

Kertotaulutehtävät: Jokaiseen ruudukon kuvakorttiin liittyy siis oma kertotauluihin liittyvä tehtävä. Oppilaat tekevät tehtävän yhdessä, tarkistuttavat aikuisella ja oikeasta vastauksesta saavat leiman leimakorttiinsa. Sen jälkeen oppilaat siirtyvät uuden ohjelmointikortin pariin.

Itsearviointilomake tehdään tunnin lopussa.

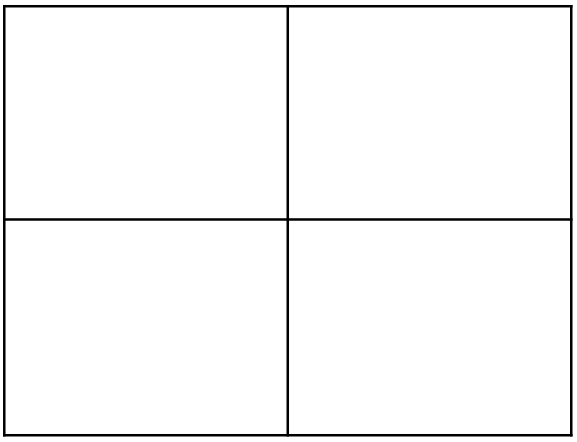
Liitteet:

Itsearviointilomake

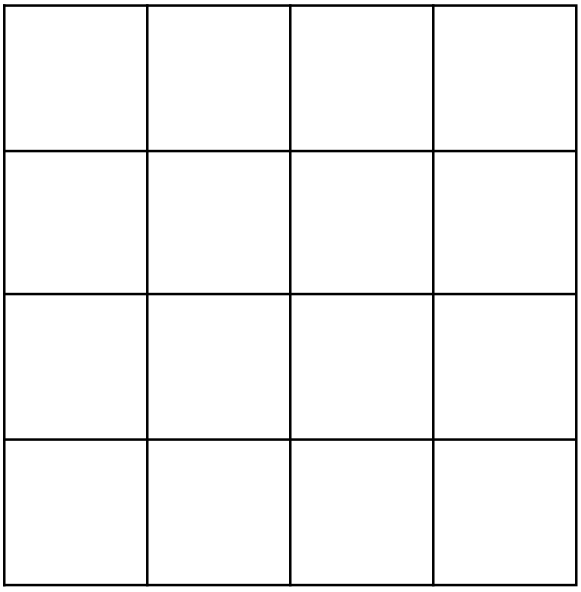
Nimi: _____

Tavoite	Itsearviointi	Open arviointi
1. Osasin työskennellä parin kanssa rauhallisesti.		
2. Keskityin tunnin tehtäviin, enkä tehnyt parin kanssa muuta.		
3. Osasin antaa parilleni ohjeita ruudukossa liikkumiseen.		
4. Osasin noudattaa parin antamia ohjeita ja pääsin oikeaan ruutuun.		
5. Muistin kertotaulut.		
6. Tunnilla oli kivaa.		

Leimakortti



Ohjelmointikortti. Piirrä reitit (ainakin 4 erilaista) etukäteen omille ruudukoille.



Kertotaulutehtävät

1. Täydennä 5-kertotaulun tulot.

1 x 5 = _____

4 x 5 = _____

2 x 5 = _____

7 x 5 = _____

8 x 5 = _____

3 x 5 = _____

6 x 5 = _____

10 x 5 = _____

5 x 5 = _____

9 x 5 = _____

2. Täydennä 2-kertotaulun tulot.

1 x 2 = _____

4 x 2 = _____

2 x 2 = _____

7 x 2 = _____

8 x 2 = _____

3 x 2 = _____

6 x 2 = _____

10 x 2 = _____

5 x 2 = _____

9 x 2 = _____

3. Muodosta kertolasku ja laske.

a. Elinalla on 4 pussia. Jokaisessa pussissa on 3 omenaa. Montako omenaa Elinalla on yhteensä?

_____ V: _____ omenaa

b. Henrillä on 6 katiskaa. Jokaisessa katiskassa on 3 kalaa. Montako kalaa Henri saa yhteensä?

_____ V: _____ kalaa

c. Siiri juoksee rataa ympäri 5 kertaa. Radalla on 4 aitaa, jotka Siiri joutuu ylittämään. Kuinka monta kertaa Siiri joutuu hyppäämään aidan yli yhteensä?

_____ V: _____ kertaa

4. Esitä kertolasku yheenlaskuna ja laske.

3 x 3 = _____ = _____

4 x 7 = _____ = _____

8 x 3 = _____ = _____

2 x 9 = _____ = _____

5 x 10 = _____ = _____

Oma tarina elämään Scratchin avulla, 3. lk

23.2.2022

Oma tarina elämään Scratchin avulla

Kohderyhmä: 3. lk, mutta sopii myös ylemmille vuosiluokille

MOK: äidinkieli, kuvaamataito, ohjelmointi

Ennakkotaidot oppilas: Oppilailla on tunnukset Scratchin käyttöön, jotta projektit voidaan jakaa. Oppilaat ovat tutustuneet aiemmin ohjelmointiin esimerkiksi ScratchJr tai BeeBotin avulla. Aiempi kokemus Scratchista ei ole pakollinen, mutta se vaikuttaa siihen, kuinka itseohjautuvasti oppilaat voivat tarinansa ohjelmoida.

Ennakkotaidot opettaja: Opettaja on tutustunut Scratchin käyttöön ja osaa mm. seuraavat asiat: hahmon ja taustan luominen, hahmon liikuttaminen, äänien lisääminen ja puhekuplien tekeminen. Perehdy tarinan luomiseen täällä: https://scratch.mit.edu/ideas → Luo tarina

Välineet: Tietokone/oppilas, oppilaiden tarinat, alkulämmittelyä varten taulu, taulutussi ja kirjainpaperit, kansio (esim. Classroomissa tai Studio Scratchissa), jonne oppilaat palauttavat linkin valmiiseen Scratch-projektiinsa.

Tavoitteet:

- Oppilas tunnistaa tarinan huippukohdan tai muun käännekohdan.
- Oppilas harjoittelee ohjeiden kuuntelua, ohjeiden noudattamista ja sinnikästä työskentelyä.
- Oppilas harjoittelee hahmon ja taustan lisäämistä Scratchissä.
- Oppilas oppii säätämään hahmon kokoa taustaan ja muihin hahmoihin sopivaksi.
- Oppilas harjoittelee hahmon liikuttamiseen tarvittavien komentojen ohjelmointia.
- Oppilas harjoittelee äänien ja puhekuplien lisäämistä.
- Oppilas testaa ja korjaa omaa projektiaan omien testien ja saamansa palautteen perusteella.
- Oppilas harjoittelee kannustavan ja rakentavan vertaispalautteen antamista.
- Oppilas arvioi omaa työskentelyään.

Palaute/arviointi:

- **Oppilas** arvioi työskentelyään mm. seuraavista näkökulmista: ohjeiden kuuntelu ja noudattaminen, töihin tarttuminen ja työssä edistyminen, avun pyytäminen ja antaminen, kannustavan palautteen antaminen, ongelmanratkaisu ja uuden oppiminen.
- **Oppilas** antaa kahdelle luokkakaverilleen kannustavaa palautetta heidän projekteistaan. Toinen oppilas valinnainen, toinen sellainen, joka ei ole saanut vielä yhtäkään palautetta.
- **Opettaja** antaa oppilaille ohjeita ja palautetta työskentelyn aikana. Opettaja kirjoittaa kommentin oppilaan tekemään itsearviointiin tai valmiiseen projektiin.

Projektia ennen: Oppilaat ovat kirjoittaneet tarinan, lukeneet ne pienissä ryhmissä ja antaneet pienessä ryhmässä palautetta kaverin tarinasta esimerkiksi piirtämällä tähden omasta mielestä parhaan kohdan kohdalle.

Projektin jälkeen/lopuksi: Vapaehtoisten tarinat voidaan lukea koko luokalle ja katsoa siihen liittyvä animointi. Seuraavaksi ohjelmointiprojektiksi sopii esimerkiksi yksinkertaisen keräilypelin ohjelmointi yksin tai parin kanssa.

Ensimmäinen oppitunti

MITÄ TAPAHTUU	TARVIKKEET	MUUTA
Aiheeseen orientoituminen leikin avulla **Kerrotaan säännöt:** Ei saa kertoa mitä tarkalleen ollaan piirtämässä. Pitää ohjata piirtäjää käyttämällä ilmaisuja, kuten ylös-alas, vasen-oikea, vinoon, kaari, keskipiste. Ilmaisut voidaan tarvittaessa kirjoittaa taululle ja opettaja voi demonstroida piirtämistä vaikka omalla etukirjaimellaan. Valitaan kirjaimien verran vapaaehtoisia piirtäjiä. Valitaan kirjaimien verran vapaaehtoisia ohjelmoijia. Ohjelmoijat kertovat vuorotellen piirtäjälleen, mitä kuuluu tehdä, käyttäen alussa yhdessä sovittuja ilmaisuja.	taulu, taulutussi, sanan kirjaimet omille papereilleen kirjoitettuna	**Lisähaaste:** Opettaja antaa kirjaimet epäjärjestyksessä ja lopuksi mietitään yhdessä, mikä sana kirjaimista syntyy. Tämä saattaa auttaa myös siinä, että oppilaat eivät voi arvata seuraavaa kirjainta sanan arvattuaan.
Animoitavien hetkien valitseminen Opettaja kertoo oppilaille, mitä projektissa tullaan tekemään. Ensimmäiseksi oppilaat valitsevat tarinoistaan kohdan, jonka herättävät eloon Scratchin avulla. Mietitään yhdessä, millainen kohta sopii animoitavaksi (tarpeeksi lyhyt, mielellään joku käännekohta). Lopuksi katsotaan yhdessä Scratchin ohjevideo tarinan animoinnista.	oppilaiden tarinat, video, Scratch auki open koneella ja älytaulu tms.	
Projektin aloittaminen Oppilaat hakevat tietokoneet, kirjautuvat Scratchiin ja liittyvät oikeaan Studioon linkin avulla. Oppilaat luovan uuden projektin. Oppilaat valitsevat tarinaansa sopivan taustan ja hahmon. Oppilaat lisäävät hahmolleen ensimmäisen puhekuplan.	oppilaiden tarinat ja tietokoneet, Scratch auki open koneella ja älytaulu tms.	Työvaiheohjeet ja opettajan ohjeet
Tunnin lopettaminen Peukkubarometri, miten työskentely sujui.		

Toinen oppitunti

MITÄ TAPAHTUU	TARVIKKEET	MUUTA
Tunnin alussa kerrataan mihin viimeksi jäätiin ja haetaan tietokoneet. Opettaja näyttää, miten lisätään muita hahmoja ja miten hahmot ohjelmoidaan keskustelemaan ennen itsenäisen työskentelyn aloittamista. Kun oppilaat ovat saaneet hahmonsa keskustelemaan, opettaja näyttää, miten hahmot saadaan liikkumaan. Oppilaat harjoittelevat hahmojen liikuttamista itsenäisesti ja opettaja kiertää auttamassa. Tämän jälkeen opettaja näyttää, miten hahmolle saadaan lisättyä ääni sitä klikatessa, jonka jälkeen oppilaat harjoittelevat äänien lisäämistä.	oppilaiden tarinat ja tietokoneet, Scratch auki open koneella ja älytaulu tms.	**Lisähaaste:** Scratchin työvaihekorteissa löytyy ohjeet myös taustan vaihtamiseen, oman puheen nauhoittamiseen ja hahmojen ulkomuodon vaihtamiseen, joiden avulla projektia voi jatkaa useamman oppitunnin ajan.
Lopetus: Oppilaat palauttavat projektinsa opettajan valitsemalle alustalle. Oppilaat antavat kannustavaa palautetta kahdelle eri projektille. Oppilaat arvioivat myös omaa työskentelyään (esim. Forms-kysely, peukkubarometri, itsearviointi paperilla tms.).	mahdolliset itsearviointilomakkeet	

ScratchJr ja liikennemerkit, 3. lk

26.4.2022

Tuntisuunnitelma:

Peruskoulun alaluokat / noin 3-luokkalaiset

ScratchJr ja liikennemerkit / ympäristöoppi ja ohjelmointi / TVT

Tavoitteet:

- Oppilas harjoittelee laatimaan algoritmeja sekä etsimään ja korjaamaan virheitä toimintaohjeissa graafisessa ohjelmointiympäristössä.
- Oppilas työskentelee sitkeästi yhteisen tavoitteen saavuttamiseksi ja ottaa vastuuta yhteisöllisestä työskentelystä.
- Oppilas jalostaa olemassa olevia ratkaisuja harjoitellen iteratiivista työskentelyä.
- Oppilas oppii tunnistamaan liikennemerkkejä.
- Erityisesti harjoitellaan ScratcJr-sovelluksessa odotustoiminnon käyttämistä ja opitaan ymmärtämään silmukan eli toistorakenteen käsitettä.

Ennen tehtävän aloittamista määritellään oppilaille/oppilaiden kanssa helposti ymmärrettävät tavoitteet ryhmän tarpeiden mukaisesti. Esimerkiksi:

- Teen työtä yhteistyössä toisten kanssa.
- Kuuntelen ja osallistun.
- Opin käyttämään silmukkaa ja odotustoimintoa.
- Opin tunnistamaan liikennemerkkejä.

Arviointi:

Tehtävä arvioidaan osana ympäristöopin sisältöjä. Formatiivinen arviointi. Opettaja havainnoi oppilaiden työskentelyä ja kirjaa muistiin havaintojaan oman työtapansa mukaisesti. Opettaja ohjaa työskentelyä rakentavan palautteen, kysymysten ja tarvittaessa konkreettisten kehittämisehdotusten kautta.

Tehtävän lopuksi käydään oppilaiden kanssa läpi tavoitteiden toteutuminen. Oppilaat voivat esimerkiksi täyttää itsearviointilomakkeen, jossa arvioidaan etukäteen määriteltyjen tavoitteiden toteutumista. Projektien esittelyn yhteydessä annetaan vertaispalautetta.

Työskentelyvälineet:

Oppilailla: iPadit/tabletit, ScratchJr-sovellus, ympäristöopin kirja tai muu lähde liikennemerkeille

Opettajalla: Ipad/tabletti, ScratchJr-sovellus, mahdollisuus näyttää ohjeita esim. älytaulun tai projektorin kautta

Opetuksen/työskentelyn eteneminen:

Ennen tätä tehtävää oppilaat osaavat jo ainakin seuraavat toiminnot ScratchJr-sovelluksessa: miten aloitetaan uusi projekti, valita hahmot/kuvata hahmoihin osia kameralla/muokata hahmoja editorilla ja tehdä liikkumiseen ja ääneen liittyviä toimintoja

Aloitus/yhteinen ohjeistus:

Opettaja esittelee tehtävän aiheen ja määritellään tavoitteet työskentelylle. Kirjoitetaan samalla taululle aihe ja tavoitteet näkyviin koko työskentelyn ajaksi.

Opettaja näyttää taululla lyhyet ohjeet/pätkän esimerkkiprojektia, jotta oppilaat saavat käsityksen siitä, mitä heidän on tarkoitus saada tehtyä ScratcJr-sovelluksessa. Esimerkkiprojektin on tarkoitus olla keskeneräinen ja erityisesti odotustoiminnoissa on huonosti toimivia aikamääreitä. Sopivien aikojen määrittäminen on

tarkoitus jättää oppilaiden oman yrityksen ja erehdyksen varaan. Kuvien esimerkkiprojektiin on mm. jätetty tarkoituksellisesti ongelmaksi peilikuvaksi kääntyvät liikennemerkit. Ohjeistuksessa keskitytään silmukan käyttöön ja näytetään esimerkki koodista myös ilman silmukka. Verrataan koodin pituutta ja huomataan silmukan hyödyt. Tässä tehtävässä koodi ei ole kovin pitkä ilman silmukkaa, mutta keskustellaan siitä, jos kysymyksiä olisikin esimerkiksi 20.

Jaetaan oppilaat pareihin. Annetaan 3 liikennemerkkiä/pari. Jaetaan pareille laitteet/annetaan lupa avata laite/sovellus.

Työskentely:

Parit työskentelevät mahdollisimman itsenäisesti. Opettaja pitää taululla esillä esimerkkiprojektia/ohjeita, jotta oppilaat voivat tarvittaessa palauttaa mieliin tehtävän ja tavoitteet. Opettaja kannustaa oppilaita kysymään neuvoa tarvittaessa myös toisilta pareilta. Opettajan roolina on toimia fasilitaattorina, kannustaa ja palauttaa oppilaat tarvittaessa ydintehtävän pariin. (Tällä kertaa kyse ei ole kuvataiteen tavoitteista, joten ei keskitytä liikaa muokkaamaan hahmojen ulkomuotoa.)

Oppilaat ohjataan valitsemaan ja luomaan projektiin 2 keskustelevaa hahmoa. Lisäksi heidän tulee luoda (kuvata kameralla) 3 heille määrättyä liikennemerkkiä hahmoksi.

Keskustelevista hahmoista toinen on kysyjä, jolle tehdään yksi äänitetty kysymys, jota toistetaan. Esimerkiksi: Mikä liikennemerkki tämä on? Kysyjälle ohjelmoidaan toiminnoiksi ääni (äänitetty kysymys) ja

odotus. Tätä toistetaan silmukan avulla kolme kertaa. Yrityksen ja erehdyksen kautta oppilaat löytävät sopivan odotusajan, jotta keskustelu toisen hahmon kanssa onnistuu. (ks. ensimmäinen kuva)

Keskustelijoista toinen on vastaaja. Hänelle ohjelmoidaan toiminnoiksi odotus ja ääni (äänitetty vastaus). Todetaan, että tässä ei silmukkarakenne toimi, koska vastaus on aina erilainen.

Liikennemerkit (3 muuta hahmoa) ohjelmoidaan odotustoiminnon avulla ilmestymään paikalle oikeaan tahtiin kysymysten kanssa. Haetaan sopiva odotus- ja ilmestymisaika yrityksen ja erehdyksen kautta.

Yhteenveto:

Kun parit saavat omat osuutensa valmiiksi, katsotaan kaikkien projektit yhdessä. Samalla kertaantuvat liikennemerkit. Opettaja antaa oppilaille kannustavaa palautetta ja ohjaa muita oppilaita kysymyksillä antamaan kannustavaa vertaispalautetta. Pohditaan esimerkiksi ajoituksen toimivuutta. Suoritetaan itsearviointi.

Tehtävän vaihtoehtoinen toteutus:

Tehtävä voidaan tehdä hieman helpompana niin, että projektiin ohjelmoidaan vain kysyjä ja tunnistettavat asiat. Vastaajan hahmo jätetään pois. Kysyjä esittää näin ollen kysymykset luokalle eikä vastauksia kerrota projektissa. Tässä versiossa kysyjä voisi kysyä lähes mitä vain, mihin vastataan lyhyesti. Esimerkiksi geometriset muodot, englannin sanat, lajintunnistus…

Karttamerkkien harjoittelua ohjelmoinnin avulla, 3. lk

30.10.2022

Kolmasluokkalaisten koodausta ja kartan käyttöä sekä karttamerkkejä yhdistävä kokonaisuus

Opetuskokonaisuuden tavoite: Opetella koodauksen alkeita samalla kun kerrataan karttamerkkejä. Osataan tunnistaa ja nimetä karttamerkit.

Oppilailla on käytössään ruudutettu kartta kuvitteellisesta paikasta ja nuolikortit (eteenpäin, käännös oikealle, käännös vasemmalle) sekä pelinappulan, josta näkyy selvästi kulkusuunta.

Alakoulun oppilaiden motivointi kaikenlaiseen pelilliseen yhteistoimintaan on yleensä helppoa, joten motivointiin riittänee mukavan näköinen kartta ja hauskat pelinappulat

Tunnin aluksi kerrataan karttamerkit, jotta oppilaat tunnistavat ne ja voivat käyttää niistä oikeita nimiä. Samalla tehdään esimerkki nuolikorttien avulla muodostettavasta koodista, jotta jokainen oppilas ymmärtäisi tehtävän idean.

Harjoituksen aluksi oppilaat harjoittelevat ruudutetulla kartalla liikkumista nuolikorttien mukaan. Toinen parin jäsen luo sovitusta aloituspaikasta alkavan reitin asettamalla nuolikortteja eteensä haluamaansa järjestykseen ja kysyy pariltaan, mihin reitti vie. Pari kulkee reitin nappulan avulla pelilaudalla ja kertoo sitten reitin suunnittelijalle, mihin reitti vie. (esim. aloituspaikkana reitillä näkyvä talo, reitti päättyi kivelle).

Reitin tekemistä voidaan harjoitella myös toisin päin: pari kertoo, mihin halutaan päästä, ja toinen parin jäsen tekee reitin korteilla pöydälle. Tämän jälkeen päämäärän päättänyt oppilas kulkee korttien mukaisen reitin, eli tarkistaa päädytäänkö reittiä käyttämällä haluttuun paikkaan.

Kun oppilaat ovat hetken harjoitelleet tällaisia tehtäviä, he voivat lisätä ohjeistukseen lisäohjeita (esim. kierrä mäen ympäri), jolloin koodeista tulee väistämättä pitempiä. Voidaan myös sopia, että reitti pitää kulkea kävellen, jolloin esim. veden yli ei saa kulkea.

Mikäli oppilas tekee väärän reitin, hän miettii reitin tekemisen jälkeen yhdessä parinsa kanssa, kuinka reitti korjataan toimivaksi.

Arviointi: opettaja arvioi ohjelmoinnin ja oppiaineen osaamista liikuskellessaan luokassa oppitunnin aikana. Oppilaat arvioivat toistensa osaamista keskusteluissaan harjoitusten aikana.

Mikäli luokassa on kilpailuhenkisiä oppilaita, voidaan tunnin lopuksi järjestää halukkaille “kuka koodaa nopeimmin”-kisa. Tällöin valitaan yksi oppilas päättämään mistä mihin reitti pitää koodata (muistetaan jälleen käyttää puheessa oikeita karttamerkkien nimiä!), ja osallistujat pyrkivät muodostamaan nuolikorteilla

mahdollisimman nopeasti oikeanlaisen koodin. Nopeimmin oikean koodin tehnyt saa pisteen. Tätä voidaan pelata ajasta riippuen yksi tai useita kierroksia.

Jos tehtävää halutaan vielä vaikeuttaa ja lisätä koodauksen osuutta, voidaan tehdä mukaan vielä “luuppikortteja” joiden avulla saman toiminnon voi tehdä useita kertoja.

Tehtävä on saanut innoitusta ja vaikutteita korttien avulla pelattavista koodauspeleistä (esim. Robogem, Roborally).

Toiminnallinen tunti ohjelmoinnillisen ajattelun harjoitteluun, 3. lk

15.10.2022

Toiminnallinen kaksoistunti ohjelmoinnillisen ajattelun harjoitteluun

Tuntisuunnitelma
LUMATIKKA 3 Algoritmisen ajattelun kehittäminen

Tämä tuntisuunnitelma on ajateltu matematiikan opetukseen, missä ohjelmointi on omana sisältönään. Tavoitteena on harjoitella ohjelmoinnillista ajattelua täsmällisten ohjeiden antamisen ja noudattamisen kannalta. Tavoitteena on huomata täsmällisten ohjeiden tärkeys toivottujen tavoitteiden saavuttamisessa sekä harjoitella näiden luomista.

Suunnittelin, että tämä voisi olla pohjatyöskentelynä ohjelmoinnillisen ajattelun harjoittelussa, josta jatkaisimme ryhmän kanssa Scratchillä ohjelmointiin. Tähän oli hyviä ohjeita jo esimerkiksi Linkin materiaaleissa (https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn=), joten en lähtenyt niitä itse luomaan, vaan ensin pohjustamaan ajattelua toiminnallisen tekemisen kautta.

Tehtävä on ajateltu omille kolmasluokkalaisilleni, mutta uskon tehtävien toimivan todella hyvin myös alkuopetuksessa ohjelmoinnillisen ajattelun harjoitteluun.

Ohjeiden antamisen täsmällisyyttä ja merkityksellisyyttä oppilaat huomaavat itse työskennellessään, sillä toimimattomien ohjeiden kohdalla ei päästä toivottuihin tuloksiin. Oppilaat työskentelevät pareittain, joten palautetta saadaan parilta. Lisäksi opettaja kiertää oppilaiden työskentelyn parissa seuraamassa ja arvioimassa oppimista. Opettaja voi haastaa oppilaita kysymällä ja kommentoimalla heidän tekemistään. Tunnin päätteeksi käydään yhteistä keskustelua oppilaiden tekemistä havainnoista tehtävien aikana.

Kaksoistunnin kulku

Ihan ensimmäiseksi, kun aihe on täysin uusi, on hyvä opettajan näyttää mallia ohjeiden noudattamisesta. Näin ollen oppilaat saavat ensiksi ohjata opettajaa. Opettaja voi ottaa roolin esimerkiksi operobottina, jota oppilaat ohjaavat tekemään haluttuja tehtäviä, esimerkiksi kirjoittamaan taululle kirjaimen A tai kävelemään luokan päästä päähän.

Tämän jälkeen siirrytään paritehtäviin. Työskentelyä voidaan toteuttaa pistetyöskentelynä tai sitten kaikki toimivat saman tehtävän parissa yhtä aikaa. Oppilaita ohjeistetaan antamaan täsmällisiä ohjeita tekemiselle ja toisaalta ohjattavia muistutetaan noudattamaan tarkasti annettuja ohjeita, vaikka tietäisikin niiden vievän tekemistä väärään suuntaan. Näin oppilaiden antamat ohjeistukset tarkentuvat ja kehittyvät tekemisen myötä.

Tehtävät ovat seuraavat:

1. **Rakennelmat.** Oppilaat saavat itse rakentaa kymmenestä annetusta multilink- tai legopalikasta oman rakennelman näyttämättä sitä parilleen. Tämän jälkeen oppilaat ohjeistavat pariaan tekemään samanlaisen pelkästään sanallisilla ohjeilla (ei siis osoittamalla eikä näyttämällä mallista). Kun rakennelma on valmis, tarkistetaan, ovatko rakennelmat samanlaisia ja keskustellaan, miksi ovat tai miksi eivät ehkä olekaan. Sitten tehdään sama tehtävä toisinpäin.
2. **Robotit.** Oppilaat ”koodaavat” toisiaan tekemään haluttuja toimintoja, kuten kirjoittamaan jonkun sanan paperille, pukemaan takin päälleen tai vaikkapa pesemään kädet. Toinen on robotti ja toinen tämän koodaaja. Tärkeää on ohjata koodaaja antamaan yksi ohje kerrallaan täsmällisesti. Robotti

kun ei pysty esimerkiksi kirjoittamaan ennen kuin hän on löytänyt kynän ja saanut sen käteensä. Annetut tehtävät voivat tulla oppilailta itseltään tai niitä voi arpoa esimerkiksi pussista nostamalla opettajan valmiiksi suunnittelemista tehtävistä.

3. **Radan kuljetus.** Opettaja on tehnyt valmiiksi jonkunlaisen liikuntaradan, joka voi sisältää esimerkiksi esteitä, eri suuntiin kulkemista, alituksia, esteiden ylityksiä, hyppimistä ja päätyä vaikkapa lattialle makaamaan. Oppilaat suorittavat tätä pareittain niin, että toinen parista antaa toiselle ohjeet, kuinka radalla kulkea eteenpäin. Ohjeet tulee antaa rauhalliseen tahtiin, jotta toinen pystyy niitä seuraamaan, ja kaikki tekeminen tulee muistaa sanoittaa. Tämän muistamista auttaa vielä se, mikäli rataa kulkevan oppilaan silmät peitetään, jolloin hän ei itse näe, mitä hänen tulisi tehdä. Tällöin toki turvallisuudesta tulee pitää erityisen tarkasti huolta ja oppilaiden pitää pystyä luottamaan toisiinsa.

Tunnin päätteeksi keskustellaan oppilaiden tekemistä havainnoista tehtävien aikana.

Ideoita sovellettu seuraavista lähteistä

Lönnroth, A. (2022) EU:n koodausviikko lokakuussa 2022. Innokas-verkosto. https://www.innokas.fi/tapahtumat/eun-koodausviikko-lokakuussa-2022/

Kangas, J. & Vartiainen, J. (toim.) (2019) Ohjelmoinnin ABC varhaiskasvatuksessa, Opettajankoulutuslaitoksen muut julkaisut, Helsingin yliopisto. http://hdl.handle.net/10138/301730

Tapahtumasarjan ohjelmointi Scratchilla, 3.–4. lk

Anna-Maija Karttunen, CC BY-SA 4.0

10.4.2021

Tuntisuunnitelma ohjelmoinnin opettamiseen 3h

Lähtötilanne: oppilaat ovat 3.-4.luokkalaisia ja he ovat kokeilleet ohjelmointia verkosta löytyvän Koodaustunti Angry Birds -tehtävien kautta. Sen lisäksi he ovat harjoitelleet matematiikan tunnilla koodausta kirjasta löytyvien tehtävien kautta.

Oppiaine: Tarkoituksena on harjoitella ohjelmointia. Halutessaan tehtävien aiheen voi ottaa jostakin oppiaineesta, esimerkiksi äidinkielen kirjallisuuden hahmojen esittely, hyvien tapojen käsittely (tervehtiminen), koska tehtävissä on tarkoitus tuoda esille vuoropuhelua.

Välineet: Tietokone tai laite, missä nettiyhteys ja mahdollisuus käyttää Scratch- ympäristöä. 1 laite/oppilaspari

Tavoitteet: Harjoitella visuaalista ohjelmointia. Tutustua koodin kirjoittamiseen ja erilaisiin käskyihin. Harjoitella yhdessä toimimista ja ongelmatilanteiden ratkaisemista.

Arviointi: Työskentelyn arviointi, jos oppilas saa ohjelmoitua pyydetyt tehtävät, on tavoitteet saavutettu. Arviointi voidaan toteuttaa seuraamalla oppilaiden toimintaa. Halutessaan opettaja voi tunnin lopuksi tehdä väittämiä ja oppilas nostaa peukun pystyyn, jos väittämä toteutui hänen kohdallaan (itsearviointi). Jos ohjelmointiin sisällytetty toisen oppiaineen sisältöjä, on syytä arvioida valmistuneen tuotoksen sisältö ja antaa siitä palautetta oppilaille.

OPPITUNTI 1

Tavoite: Tutustutaan Scratch-ympäristöön.

a) Aluksi harjoitellaan menemään Scratch-sivustolle ja harjoitellaan kirjautumaan sisälle, jos oppilailla on tunnukset. (https://scratch.mit.edu/)
b) Opettaja näyttää keskeiset asiat sivuston toiminnasta:

- sivuston osat (koodien paikka, koodaustila ja esiintymislava)
- hahmon valitseminen/vaihtaminen/ poistaminen/ koon muuttaminen
- taustan valitseminen/vaihtaminen
- koodien värien idea ja miten löytää sopiva koodi helpommin
- aloituspalikka, koodin kokeilun käyntiin laittaminen + pysäyttäminen

c) Tämän jälkeen oppilaat kokeilevat ympäristön toimintaa pareittain itsenäisesti, koska yleensä yleensä oppilaiden into on tässä vaiheessa todella korkealla ja he keksivät itse vaikka mitä kivaa.

OPPITUNTI 2

Tavoite: Koodataan lyhyt tapahtumasarja yksinkertaisilla komennoilla, jotka koskevat yhtä hahmoa. Toimitaan pareittain.

1. Valitse jokin hahmo ja tausta.
2. Koodaa hahmo kulkemaan esiintymislavan reunasta toiseen. (esim. liukumalla)
3. Koodaa hahmo tervehtimään.
4. Koodaa hahmo kulkemaan takaisin lähtöpaikkaansa.
5. Koodaa hahmo sanomaan tämän jälkeen “Olipa reissu”.
6. LISÄTEHTÄVÄ:
   - koodatkaa hahmo katoamaan, kun “Olipa reissu” on sanottu.
   - koodatkaa, että yllä olevat kohdat 1-4 toistuu loputtomasti.

- kehitä koodia eteenpäin haluamallasi tavalla

OPPITUNTI 3:

Tavoite: Koodata lyhyt tapahtumasarja, jotka koskevat kahta hahmoa. Toimitaan pareittain

1. Valitse 2 hahmoa ja niille sopiva tausta.
2. Toinen hahmo menee toisen luo ja kysyy tältä jotain.
3. Tämän jälkeen toinen vastaa ja lähtee pois.
4. LISÄTEHTÄVÄ:
    - lisätkää vuorokeskustelua hahmojen välille
    - lisätkää esiintymislavalle jokin asia, joka toistuu koko ajan esim. jokin eläin liikkuu taustalla, lunta sataa…
    - kehitä koodia haluamallasi tavalla

Koordinaatiston harjoittelu laivanupotus-pelin avulla sekä Scratch-projekti, 3.-4. lk

Maija Takalo, CC BY-SA 4.0

4.11.2022

TUNTISUUNNITELMA OHJELMOINNIN OPETUKSEEN

Koordinaatiston harjoittelu laivanupotus- pelin avulla sekä Scratch-projekti

3.–4.luokka

Maija Takalo

Aihepiiri sekä siihen liittyvät oppimistavoitteet ja arviointi

- Ohjelmointi liitetään osaksi matematiikan oppitunteja, koska tämä tuntuu ohjelmointia opettelevalle opettajalle eli minulle luonnolliselta ennen isompiin monialaisiin projekteihin ryhtymistä. Valitsin Scratch-projektiksi näin aluksi keräilypelin, koska se on tämän kurssin myötä itselleni nyt tuttu. Tämän kurssin keräilypelin ohjeen lisäksi aion laajentaa peliä oppilaiden kanssa.
- Oppituntien tavoitteena on tutustua ja harjoitella oppilaiden kanssa koordinaatistoa ja sen käyttöä sekä tutustua Scratch-ohjelmointiympäristöön. Pelillisyys molemmissa osioissa todennäköisesti motivoi oppilaita.
- Arviointi perustuu jatkuvan havainnoinnin periaatteisiin. Opettaja ohjeistaa sekä ohjaa ja oppilaat saavat itse kokeilla sekä oppia. Opettaja kannustaa myös vertaistensa auttamiseen, josta opettaja voi myös saada vinkkiä, jos luokassa on harrastuneisuutta/aikaisempaa osaamista ohjelmoinnin osalta jollakin oppilaalla. Opettaja antaa kohdennettua palautetta oppilaille oppituntien kuluessa. Lisäksi ohjelmointiympäristö antaa oppilaille välitöntä palautetta ohjelmoinnin onnistumisesta, jonka kautta oppilaan on myös mielekästä toteuttaa itsearviontia.

Työskentelyvälineet

- ruutupaperia ja kyniä laivanupotus-peliin

  Opettaja kopioi myös valmiiksi muutamia koordinaatistoja niitä oppilaita varten, joiden on vaikea saada itse koordinaatistoa piirrettyä mielekkäässä ajassa. Jokainen oppilas yrittää kuitenkin ensin itse.

- Chromebookit pareittain tai jokaiselle oppilaalle oma. Opettajan on pitänyt muistaa varata koulun chromekärry.
- Luokan esitysgrafiikka; dokumenttikamera, tietokone ja tykki, jonka kautta opettaja antaa ohjeistuksen.
- Valmiiksi tulostetut ohjeet oppilaille keräilypelistä. Linkissä ohje: https://www.cs.helsinki.fi/group/linkki/materiaali/lapsiohjeet/kerailypeli/kerailypeli.pdf

Opetuskokonaisuuden kulku

Luulen, että opetuskokonaisuuteen tarvitaan 2–3 oppituntia tai tarvittaessa jopa enemmän. Tässä suunnitelmassa en erottele oppitunteja toisistaan, vaan kokonaisuuden voi toteuttaa ryhmästä riippuen sopivassa aikataulussa.

Ensimmäisellä oppitunnilla tutustutaan koordinaatistoon laivanupotuspelin avulla. Opettaja esittelee oppilaille koko koordinaatiston havainnekuvan, jos se on heille uusi asia, mutta kertoo sen jälkeen, että tänään työskennellään vain yhden neljänneksen koordinaatiston kanssa. Oppilaat pelaavat peliä pareittain. Opettaja jakaa oppilaille ruutupaperia, johon he piirtävät mallin mukaan koordinaatiston yhden neljänneksen sekä koordinaatistoon kahdeksan laivaa. Opettaja näyttää Pulmaario-paketista havainnekuvan. Halutessaan peliä voi laajentaa myös koordinaatiston puolikkaaseen tai koko koordinaatistoon. Kun oppilailla on laivat piirretty, peli alkaa ja parit yrittävät upottaa toistensa laivoja. Laivanupotuspeli tutustuttaa oppilaat koordinaatistoon sekä samalla he harjoittelevat koordinaatiston hahmottamista, jota tarvitaan myöhemmin ohjelmoidessa. Lisäksi laivanupotuspeli vaatii oppilailta huolellisuutta laivojen merkitsemisessä sekä yhteistyötaitoja pelitilanteessa. Peliä voidaan pelata uudestaan oppilaiden innostuessa siitä.

(Lähde: Pulmaario, s. 38 -40)
https://www.cs.helsinki.fi/group/linkki/materiaali/pulmaario/ohjaajanopas/Pulmaario_ohjaajanopas.pdf)

Laivanupotuspelin jälkeen, kirjaudutaan koulun Chromebookeille ja luodaan tunnukset Scratchiin opettajan luoman linkin kautta, joka on lähetetty oppilaille sähköpostiin tai vaihtoehtoisesti opettaja on tehnyt tunnukset oppilaille valmiiksi. Tässä ohje tunnusten luontiin, jota ajattelin itse kokeilla myöhemmin:
https://peda.net/joensuu/jm/oio/alakoulu/ojr/loay/scratch/opettajatili

Tämän jälkeen opettaja näyttää oppilaille mallipelin keräilypelistä, jotta oppilaat hahmottavat, mitä ollaan tekemässä. Opettaja on tehnyt itse mallipelin tai esim. seuraavaa linkkiä voi käyttää apuna: https://scratch.mit.edu/projects/56580404/ . Mallipelin jälkeen oppilaille jaetaan keräilypelin tulostetut ohjeet. Pelin tekeminen aloitetaan opettajajohtoisesti, jotta kaikki pääsevät ideaan kiinni. Kun opettaja huomaa, että oppilaat alkavat hoksata Scratchin toimintaidean, voivat he jatkaa itsenäisesti työskentelyä ohjeen mukaan eteenpäin. Olisi hyvä, jos opettaja olisi etukäteen ehtinyt itse toteuttaa pelin, jotta olisi helpompi auttaa oppilaita. Seuraavan linkin takaa löytyy ohje komennoista ja opasvideo tarvittaessa opettajalle, joiden avulla hän pystyy ohjaamaan oppilaita oikeaan suuntaan kuitenkin sammuttamatta oppilaiden omaa oivaltamisen iloa. Tässä linkki:
https://www.cs.helsinki.fi/group/linkki/materiaali/kirjastopaja/Kerailypeli.html

Ohjelmointitehtävää voi eriyttää ylöspäin laajennusideoilla sekä vastaavasti alaspäin yksinkertaistamalla peliä, jos tarve vaatii. Lisäksi oppilaille voi tarjota lisähaastetta tarjoamalla erilaisia pelejä näyttämällä esimerkkejä esim. Helsingin yliopiston linkki-sivustolta: https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn= . Saman pelin ohjelmoiminen jakautuu todennäköisesti useammalle oppitunnille, joten peli täytyy välillä muistaa tallentaa, jotta tämä onnistuu.

Ohjelmoinnin aikana opettaja voi pyytää oppilaita pohtimaan koordinaatiston ja ohjelmoinnin yhteyttä ja saada heidät itse oivaltamaan yhteyden. Ohjelmoinnin eri vaiheissa, olisi mielestäni mielekästä käydä oppilaiden kanssa keskustelua myös seuraavista asioista:

- Mihin ohjelmointia oppilaiden mielestä tarvitaan? Onko ohjelmoimisen osaaminen hyödyllinen taito?
- Millaisia tunteita ohjelmointi herättää oppilaissa? Miksi?
- Mitä he ovat oppineet? Mikä on helppoa, mikä vaikeaa?
- Millaista ohjelmointikieltä Scratchissa esiintyy?

Valmiit pelit on lopuksi mielekästä esitellä koko luokalle tai jopa rinnakkaisluokalle niin, että ainakin jokainen ohjelmointiin osallistunut oppilas pääsee testaamaan toisen oppilaan luomaa peliä.

Lähteet:

Helsingin yliopsto, linkki-sivusto
https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn= (luettu 4.11.2022)

Pulmaario – matematiikkaa ja ohjelmointia, ohjaajan opas
https://www.cs.helsinki.fi/group/linkki/materiaali/pulmaario/ohjaajanopas/Pulmaario_ohjaajanopas.pdf (luettu 4.11.2022)

Joensuun mediakeskus, Peda.net
https://peda.net/joensuu/jm/oio/alakoulu/ojr/loay/scratch/opettajatili (luettu 4.11.2022)

Ohjelmoinnin alkeita: BeeBot ja Code.org, 3.–6.lk

14.2.2021

Tuntisuunnitelma: pienryhmä, oppilaat luokilta 3-6, monenlaisia haasteita sosiaalisissa- ja kongnitiivisissa taidoissa, matematiikassa suuria haasteita.

Aihe: Ohjelmoinnin alkeita: toisen ohjaaminen, beebot ja code.org

Kohderyhmä: pienryhmä 3.-6.-luokka, oppilaita 8, oppimistaso ei ole luokkatasojen mukainen

Oppiaineet: matematiikka

- Mietin aina oppitunneille enemmän tehtäviä kuin todennäköisesti ehditään käymään läpi. Tämä siitä syystä, että oppilaat muuttuvat heti levottomaksi, jos opettajalla ei ole valmiina seuraava tehtävä, tai välipalatehtäviä eli odotukset eivät onnistu tämän ryhmän kanssa.

- Ohjelmointi on minulle ja suurimmalle osalle oppilaista uutta. Olemme jonkin verran käyttäneet Bee bot-robotteja eri oppiaineissa. Varsinainen tietokoneella ohjelmointia on kokeillut kaksi kuudesluokkalaista.

- Teemme tehtävät kahdessä ryhmässä 3.-4. luokkalaiset (käyttävät enemmän aikaa kahteen ensimmäiseen tehtävään) ja 5.-6.-luokkalaiset, jotka käyvät nopeasti läpi kaksi ensimmäistä tehtävää ja siirtyvät code.org- sivustoon

- Tunnit ovat suunniteltu toiminnalliseksi, näin oppilaat toivon mukaan motivoituvat tehtäviin.

- Kaksi ensimmäistä tehtävää tehdään parin kanssa.

- Kerrataan käsitteitä: vasemmalle, oikealle, eteen, taakse

Ennakkovalmistelut:

· Bee boteille pohja ruudukko, joissa kertolaskujen vastauksia, tehdään oppilaiden kanssa

· Kertolaskukortit oppilaan ohjaamiseen ja bee boteille

· Kertolaskujen vastausympyrät käytävälle

· Ohjeet code-org sivuille pääsyyn

- Multilink-kuutioita, pöytäsermit, ruutupaperit, lyijykynät.

Tavoitteet:

· Harjoitellaan käskyjen antamista ja noudattamista opettajan ohjaamana ja toisen oppilaan ohjaamana.

· Harjoitellaan suuntia ja paikan määreitä, ja niiden käyttämistä ohjeiden antamisessa.

· Opitaan antamaan selkeitä ohjeita ja oppia noudattamaan niitä.

· Herätetään mielenkiinto ohjelmointiin.

· Kerrataan kertolaskuja.

Paikka: - Oma kaksiosainen luokka sekä käytävä

§ Käytävä: kaverin ohjelmointi

§ Isompi luokka: Bee bot harjoitukset

Aika: Kaksi peräkkäistä oppituntia

Välineet:

- Bee bot- robotit 4kpl

§ Pienempi luokka: Tietokoneet

4 alustaa, joihin opettaja on merkinnyt kertolaskujen vastauksia, pienemmille oppilaille tarkistuspaperit

- Neljä päätepistettä (isot värikkäät kartongista tehdyt ympyrät, joissa kertolaskun tuloksia) käytävälle.

- Kertolaskukortit

- 8 tietokonetta

- Taululla kirjautumisohjeet code.org-sivuille

- Tulostin – code.org todistusten tulostusta varten

Arviointi:

· Oppilaat arvioivat keskustelussa omaa oppimistaan, mitä uutta opittiin, mitä ohjelmointi on jne.

· Code.org sivuilta tulostetaan oppilaille seinälle salanimellä varustetut todistukset.

· Kysellään kertolaskuja

· Opettaja ja ohjaaja tarkkailevat oppilaiden toimintaa ja antavat kannustavaa ja motivoimaa palautetta sekä keskustelevat oppilaiden kanssa tehtävistä.Kummankin ryhmän kanssa kulkee opettaja tai ohjaaja mukana antamassa palautetta ja huomioimassa milloin vaihdetaan tehtävää.

Oppitunnit 2h

Oppilaan ohjaaminen käytävällä kahdella pisteellä

· Aloitetaan tunti opettajan ohjeita noudattaen. Opettaja kertoo ohjeet kaikille oppilaille yhteisesti muutaman kerran, kunnes jokainen tietää mitä pitää tehdä.

· Oppilaat jaetaan pareihin, joissa toinen pareista antaa ohjeet ja toinen toimii ohjeiden mukaan. Eli ohjeita antava oppilas nostaa kertolaskukorista yhden laskun, etsii käytävän seinältä ohjeen ja antaa ohjeita parille kertolaskun vastauksen luo. Sovitaan etukäteen, että ohjeita pitää antaa vähintään viisi eli ei vain esim. suoraan 5 askelta ja vasemmalle 3 askelta. Askeleet ovat lattiassa olevien ruutujen mittaisia. Vuoron vaihto.

Bee boteilla kertolaskuja isossa luokassa

· Opettaja/ohjaaja kertaa robottien ohjaamisen. Tämän jälkeen oppilaat jakaantuvat pisteille pareittain.

· Nostetaan kertolaskukortti ja ohjataan vuorotellen robotti oikean vastauksen luo.

Code.org tietokoneella

· Käydään yhdessä monisteen avulla vielä ohjeiden antamista. Ensimmäisessä ohjeessa ei ole esteitä, ohjaamme vain madon kotipesään. Toisessa monisteessa on esteitä, muuten sama tehtävä.

· Käydään taululta läpi, miten päästään ohjelmaan.

· Oppilaat menevät ohjelmaan ja ottavat ensimmäisen Angry birds tehtävän. Jokainen saa edetä omaa vauhtia. Kun koko tehtävä sarja on käyty läpi tulostetaan

oppilaan itse keksimällä salanimellä todistus luokan seinälle. Tehtäviä ei ole tarkoitus saada yhdellä kertaa valmiiksi.

· Opettaja/ohjaaja tarkkailee ja ohjaa tarvittaessa.

Varalle lisätehtävä

· Multilinkeillä rakentaminen

- Pikkusermi oppilaiden välissä. Molemmilla yhtä monta ja samanväristä multilik-kuutiota. Toinen pareista rakentaa opista multilinkeista jotain ja antaa parille ohjeita sermin takaa. Pari yrittää rakentaa samanlaisen rakennelman kuulemiensa ohjeiden mukaan.

· Piirrä samanlainen

- Pareilla samanlaiset ruutupaperit (isot ruudut) ja lyijykynät. Ruutupapereissa lähtöpiste on merkitty. Toinen pareista piirtää pikkusermin takana jonkin kuvion ruutupaperilla ja antaa ohjeita parille, joka koettaa piirtää samanlaisen kuvion kuulemiensa ohjeiden mukaan. Ohjeet pitää kulkea ruutujen mukaan esim. kaksi ruutua vasemmalle, kolme ruutua ylös jne.

· Robogem lautapeli

Välipalaksi: Kapteeni käskee (ohjeet koodauksen mukaisesti)

Visuaalisen ohjelmoinnin harjoittelu, 3.–6. lk

8.5.2022

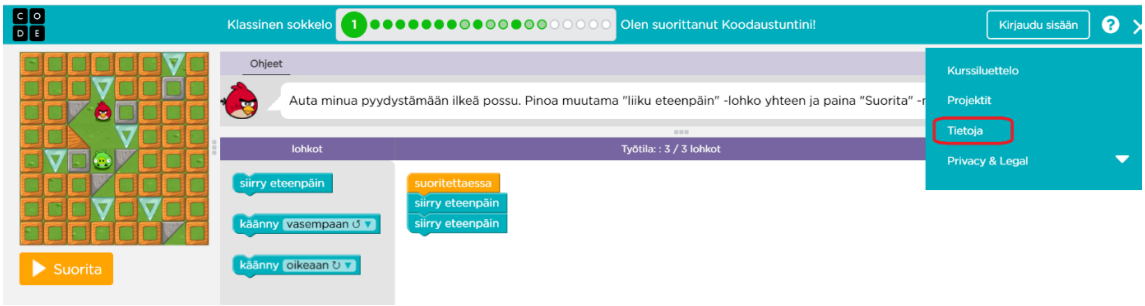
Tuntisuunnitelma

Matemattiikka, 3lk-6lk, visuaalisen ohjelmoinnin harjoittelu

Alkuvalmistelut

Opettaja tutustuu ohjelmointiympäristöön ja käy halutessaan kirjautumassa sisään sivustolle Classic Maze - Code.org.

Ohjelman kielen saa valittua “Tietoja” -valikon alta.



Ohjeita opettajalle code-ohjelman käytöstä:

https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start-

Ohjelmaa voi käyttää ilman tunnuksia, mutta silloin tehdyt harjoitukset eivät tallennu mihinkään, eikä opettaja voi seurata oppilaiden edistymistä. Tämän tunnin ajatuksena on opetella algoritmista ajattelua. Tehtäviä ei ole pakko tallentaa. Jos haluat tehdä tunnukset, voit sen jälkeen luoda omia luokkahuoneita ja voit seurata jokaisen oppilaan edistymistä tehtävissä. Kirjautuneena kaikki työt voidaan myös tallentaa. Valitse itse, kumpi tapa toimii sinulle paremmin.

Tunnin kulku

Tällä tunnilla harjoitellaan algoritmista ajattelua. Tunnin alussa katsotaan videoita (samat videot tulevat uudestaan harjoitusten lomassa, mutta se ei haittaa, koska asiaa kannattaa kerrata). Ne ovat englanninkielisiä, joten niistä kannattaa keskustella, jotta kaikki ymmärtävät pääpointin. Jos englanninkieliset videot eivät toimi ryhmäsi kanssa, voit jättää videon katselut pois ja siirtyä suoraan esimerkkeihin.

Tässä linkit videoihin:

1. (kesto 2:01) https://www.youtube.com/watch?v=bQilo5ecSX4
2. (kesto 1:34) https://youtu.be/mgooqyWMTxk
3. (kesto 0:50) https://youtu.be/G2hdlhDYICw

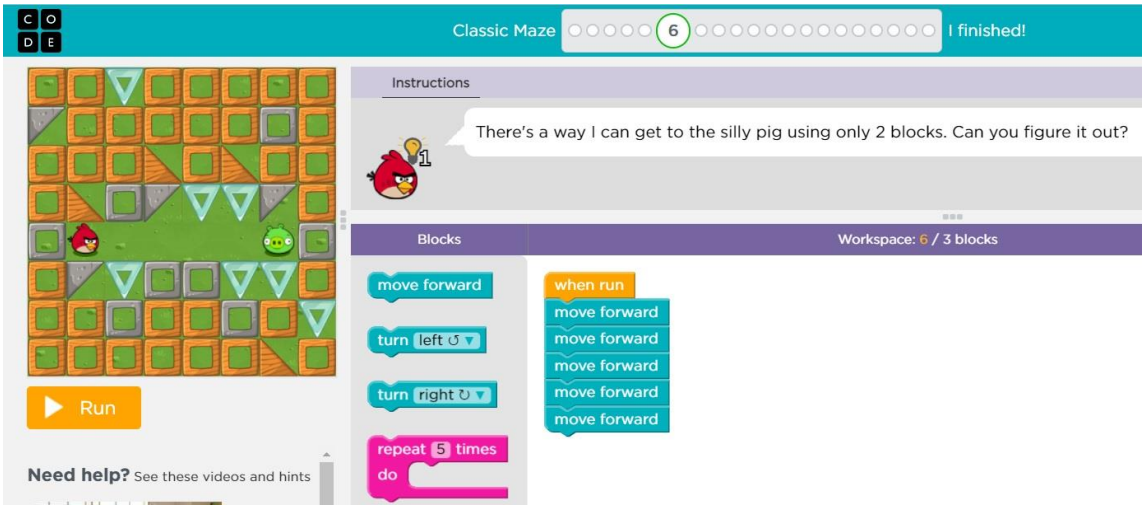
Tämän jälkeen katsotaan opettajan johdolla yhdessä muutaman esimerkin avulla, miten ohjelma toimii. Sen jälkeen oppilaat saavat itse valita parin, kenen kanssa he alkavat tehdä harjoituksia.

Video 1 jälkeen harjoituksia (5-10min)

Video 2 jälkeen harjoituksia (5-10min)

Video 3 jälkeen harjoituksia lopputunti

Ohjelman käyttöä voit näyttää esimerkiksi tehtävän nro 6 avulla:



Keltaisella "Run" -napilla saa ajettua ohjelmaa.

"Blocks" -osiosta saa hiirellä raahattua ohjelmointipalikat valkoiselle "Workspace" -alueelle.

Harjoitukset löytyvät: https://studio.code.org/hoc/1 -sivustolle.

Aihepiirin valinta ja rajaus

Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Matematiikan tunnilla. 3-6luokkien matematiikan opetussuunnitelmassa opetuksen tavoitteisiin S1 ajattelun taidoissa kuuluu, että oppilaan pitää osata suunnitella ja toteuttaa ohjelmia jossakin graafisessa ohjelmointiympäristössä. Työkaluna toimii visuaalinen ohjelmointiympäristö, jossa voidaan työskennellään hiiren avulla, eli ei tarvitse osata kirjoittaa ohjelmia. Tällä tunnilla harjoitellaan ohjelmien tekemistä laittamalla valmiita koodipalikoita oikeaan järjestykseen ja ajamalla sen jälkeen ohjelma. Oppilas näkee heti, onko koodi oikein vai väärin ja voi kokeilla niin monta kertaa uudestaan, kuin on tarvetta.

Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Näillä tehtävillä harjoitellaan algoritmista ajattelua. Myös täytyy laskea, montako askelta otetaan ja minne suuntaan (oikea, vasen, ylös, alas), vaikeamissa tehtävissä on jo silmukoitakin (toistoa) mukana.

Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

-Oppilas hahmottaa käsitteet oikea, vasen, ylös, alas

-Oppilas osaa laittaa toimintoja oikeaan järjestykseen, mitä pitää tehdä ensin, jotta päästään seuraavaan kohtaan esimerkiksi ensin pitää antaa suuntakäsky (oikea, vasen, alas, ylös) ja sen jälkeen vasta annetaan käsky, montako askelta.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

-Tavoitteena on, että oppilaat saisivat tehtyä harjoituksia mahdollisimman paljon ilman opettajan apua

-Suuntakäsitteet tulevat rutiiniksi (oikea, vasen, ylös, alas)

-Luetun ymmärtäminen

-Oppilaat oppivat hahmottamaan, missä järjestyksessä käskyt pitää antaa, jotta ohjelma toimii halutulla tavalla

Miten opettaja tai joku muu arvioi opiskelijan osaamista? Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Arvioidaan kokonaisuutena. Tärkeää näissä tehtävissä on herätellä innostusta ohjelmointiin ja saada oppilas kiinnostumaan aiheesta.

Opettajalla on mahdollisuus kirjautua ohjelmaan sisään ja tehdä sinne oma ryhmä. Tällöin opettaja voi seurata oppilaiden tehtävien suoritusta ja tehtäväpisteet tallentuvat.

Miten oppilas arvioi omaa osaamistaan?

Oppilaat saavat tunnin lopuksi kertoa, miltä ohjelmointi tuntui. Mikä oli helppoa? Mikä oli vaikeaa? Olisiko halunnut tehdä yksin? Oliko kiva tehdä parin kanssa?

Työskentelyvälineet ja opetusmenetelmät

Mitä välineitä oppikokonaisuudessa käytetään?

Työvälineenä tietokone tai tabletti

Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Tunnin alussa katsotaan yhdessä opettajan johdolla, miten ohjelma toimii, mistä löytyy mitäkin ja muutamia esimerkkejä.

Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Tehtäviä saa yrittää ratkaista niin monta kertaa, kuin on tarve. Tehtävät muistuttavat pelin pelaamista. Ympäristö on värikäs ja selkeä.

Miten opetuksessa rohkaistaan yhteistyötä?

Harjoitukset tehdään parin kanssa. Välillä vaihdetaan sitä, joka istuu koneella.

Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

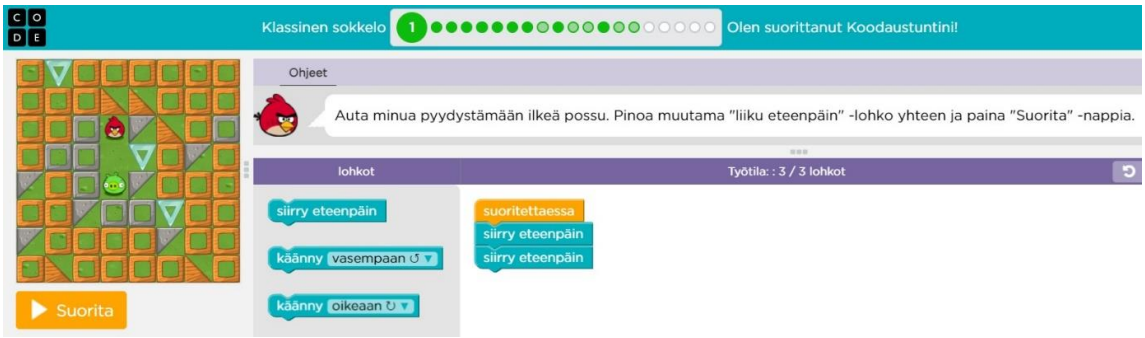
Aiempi osaaminen näkyy varmasti siinä, miten nopeasti oppilaat saavat tehtävät ratkaistua.

Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelua?

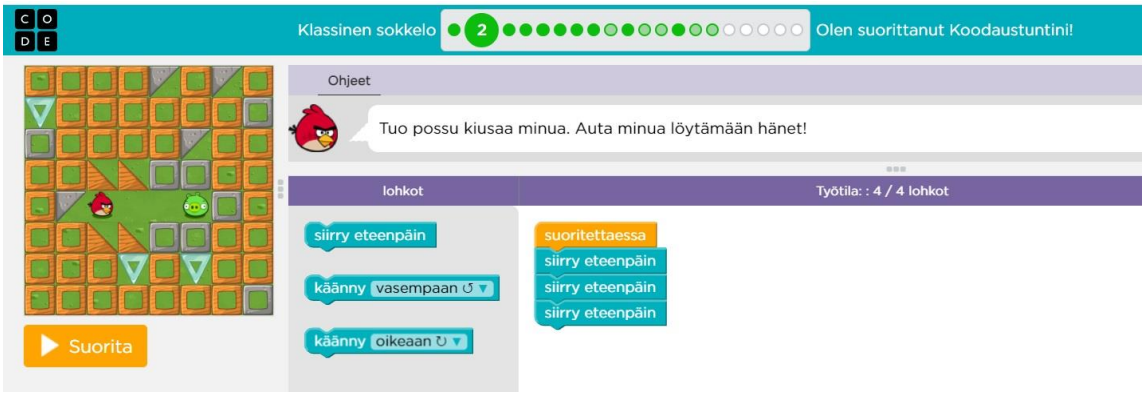
Näiden harjoitusten jälkeen voidaan siirtyä kokeilemaan ohjelmointia ScratchJR:llä tai Scratchillä riippuen ryhmän osaamisesta. ScratchJR on visuaalinen ja Scratchissä on jo tekstiäkin mukana.

VALMIIT ESIMERKKIVASTAUKSET OPETTAJALLE TEHTÄVÄT 1-14

Tehtävä 1



Tehtävä 2



Tehtävä 3

Tehtävä 4

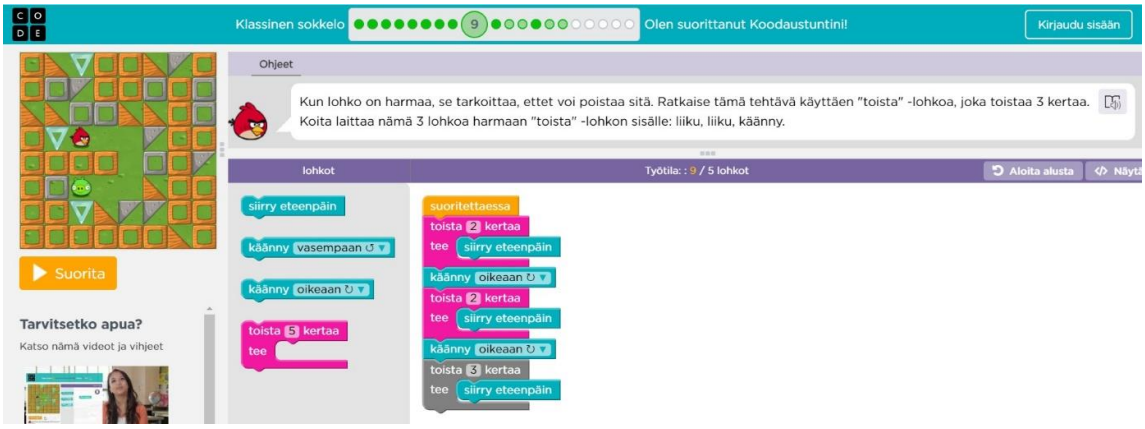
Tehtävä 5

Tehtävä 6

Tehtävä 7

Tehtävä 8

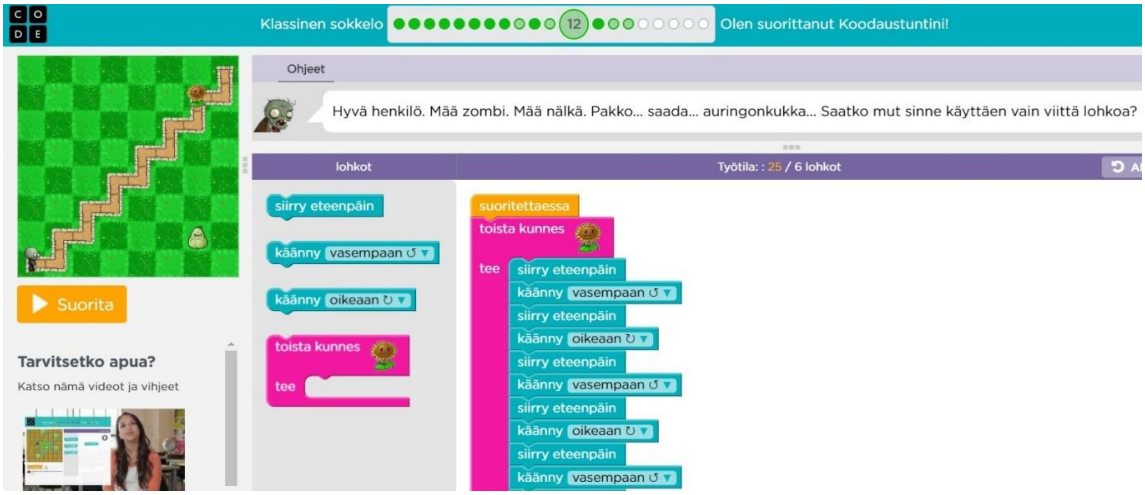
Tehtävä 9



Tehtävä 10

Tehtävä 11

Tehtävä 12



Tehtävä 13

Tehtävä 14

LÄHTEET:

https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start-

https://www.youtube.com/watch?v=bQilo5ecSX4

https://youtu.be/mgooqyWMTxk

https://youtu.be/G2hdlhDYICw

https://studio.code.org/hoc/1 -sivustolle

https://www.raahe.fi/sites/raahe.fi/files/liitetiedostot/Kasvatus%20ja%20koulutus/OPS_2016_0.pdf

Koodilukko: Ongelmanratkaisua silmukalla ja listalla, 3.–6. lk

Teemu Korhonen, CC BY-SA 4.0

14.9.2022

Tuntisuunnitelma ohjelmoinnin opetukseen

Koodilukko: Ongelmanratkaisua silmukalla ja listalla

Laatija: Teemu Korhonen

Oppitunnin tavoitteet

Tällä oppitunnilla oppilaiden tehtävänä on **ohjelmoida kone, joka selvittää lukujen jaollisuutta**. Jaollisuuden tutkiminen on matematiikan oppisisältönä kaikilla vuosiluokilla 3-6. Ohjelman kirjoittaminen edellyttää oppilailta jaollisuuden käsitteen ymmärtämistä (luku a on jaollinen luvulla b, jos a/b menee tasan eli ei jätä jakojäännöstä), mutta ohjelmaa voidaan valmiina käyttää jaollisuuden tutkimiseen. Tätä varten on esitetty tuntisuunnitelman lopussa muutamia pohdintakysymyksiä, opettaja/oppilas voi keksiä itse lisää.

Ohjelmoinnin ja algoritmisen ajattelun puolelta **tällä tunnilla harjoitellaan/tutustutaan Scratch-ympäristöä käyttäen ohjelmoinnin käsitteisiin**

- muuttujat
- ehtolause (jos-niin; jos-niin-muuten)
- silmukka (toista, kunnes…)
- listat (listat vain vaikeammassa tuntisuunnitelmassa)

Tunnin **tavoitteena on osoittaa oppilaille myös se, että on asioita, joita tietokoneohjelma kykenee tekemään huomattavasti ihmistä tehokkaammin, antaa oppilaille kokemus tällaisesta tilanteesta sekä saada oppilaat pohtimaan muita arkielämän tilanteita, joissa kone voi olla tehokas.**

Työskentelyn tavoitteina on

- harjoitella pitkäjänteistä ja sinnikästä työskentelyä
- harjoitella pari- tai ryhmätyöskentelyä: omien ideoiden sanallistamista, toisen kuuntelua, yhteistyössä toimimista
- harjoitella tehokasta ohjelmointia ja ongelmanratkaisua: esimerkiksi ongelman ymmärtäminen, sen sanallistaminen ja visualisointi esim. taululle, ongelman muuttaminen ohjelman vaatimuksiksi, sen pilkkominen osaongelmiin ja ongelman ratkaiseminen vaiheittain osaongelmien kautta

Kohderyhmä ja ennakkotietovaatimukset

Oppilailla tulisi olla ymmärrys jaollisuuden merkityksestä. Tuntisuunnitelma ei ole rajattu tietylle luokka-asteelle, vaan sen soveltuvuus riippuu oppilaiden taidoista. Suunnitelmassa on kaksi eritasoista tehtävää, joista helpompikin edellyttää perusosaamista Scratchissa. Ei siis aivan aloittelijoille.

Ennen tehtävän suorittamista parin tai koko luokan kanssa voidaan yhdessä luokan kanssa tutustua muuttujan ja listan käsitteeseen ja selvittää, miten nämä Scratchissa toimivat, mikäli nämä ovat uusia asioita.

Muuttujan käsitettä voi havainnollistaa oppilaille esimerkiksi piirtämällä taululle kaksi laatikkoa, jotka kuvaavat muuttujia. Tämän jälkeen laatikot nimetään kuvaavilla nimillä. Nimet voisivat olla esimerkiksi ”jaettava” ja ”jakaja”. Kirjoitetaan sitten juuri luotujen muuttujien nimityksiä käyttäen taululle matemaattisia lausekkeita. Esimerkkisi jaettava+jakaja, jaettava*jakajaope jne. Heitetään kahta noppaa ja sovitaan, että ensimmäinen antaa arvon muuttujalle ”jaettava” ja toinen muuttujalle ”jakaja”. Lasketaan lausekkeiden arvot ja heitetään noppaa uudelleen. Näin huomataan, että lauseke on ikään kuin toimintaohje, jonka arvo muuttuu sen mukaan, mitkä muuttujat kirjekuorista tai laatikoista löytyvät.

Samaan harjoitukseen voidaan yhdistää listoihin tutustuminen piirtämällä taululle esimerkiksi kaksi listaa. Toisen nimi voi olla esimerkiksi ”testattava luku” ja toisen ”parilliset luvut”. Testattava luku -listalle kerätään lukuja, joista ensimmäinen testataan jakamalla se kahdella. Jos jako menee tasan, luku on parillinen, ja luku siirretään parilliset luvut-listalle. Jos jako ei mene tasan, luku poistetaan listalta. Tämän jälkeen otetaan aina seuraava luku listalta ja toistetaan sama.

Arviointi

Opettaja voi itse päättää kohderyhmän aiemmasta ohjelmointikokemuksesta riippuen, arvioidaanko tässä tehtävässä oppilaiden taitoa käyttää em. ohjelmointikäsitteitä. Jos nämä käsitteet ovat uusia, arvioinnin kohteena voi olla myös työskentely- ja ongelmanratkaisutaidot. Katso yllä työskentelyn tavoitteet ja muokkaa ne ryhmällesi sopiviksi kriteereiksi.

Kerro oppilaille etukäteen, mihin tässä tehtävässä arvioinnin osalta kiinnitetään huomiota. Tee kriteereistä vaikkapa kolmiportainen asteikko: 1. Emme tehneet yhteistyötä ollenkaan. 2. Teimme osittain yhdessä, mutta myös aika paljon yksin. 3. Keskustelimme tehtävästä ja ratkoimme ongelmaa yhdessä alusta loppuun. Näin oppilaat tietävät, mitä tehtävässä tavoitellaan voivat itsekin arvioida, miten onnistuivat. Oppilaiden on esimerkiksi hyvä tietää, että tämä tehtävä ei ole kilpailu, jossa nopein voittaa.

Toteutus

Tämä suunnitelma pitää sisällään itse asiassa kaksi eritasoista suunnitelmaa. Opettaja voi oppilaiden taitotasosta riippuen päättää, tekeekö vain helpomman tai vaikeamman. Mikäli molemmat tehdään, voidaan toteutus jakaa kahdelle tunnille niin, että ensin tehdään helpompi ja toisella tunnilla esitellään jäljempänä kerrottu numerokoodilukkohaaste, jonka ratkaisuun oppilaat (toivottavasti) itsekin oivaltavat tarvitsevansa monipuolisempaa ohjelmaa.

Tuntisuunnitelma on toteutettu Scratch-ympäristöön, mutta sitä voi soveltaa myös esimerkiksi Python-kielellä tehtäväksi tekstiympäristössä.

Ensimmäisessä, helpommassa versiossa oppilaiden tehtävänä on laatia ohjelma, joka kysyy käyttäjältä kahta lukua, jaettavaa ja jakajaa, ja ilmoittaa sitten onko jakaja jaollinen jaettavalla. Tämä on ihan mukava perusohjelmointi- ja ongelmanratkaisuharjoitus, mutta oikeastaan ohjelmasta ei ole mitään käytännön hyötyä, sillä jos oppilas ymmärtää jaollisuuden tarkoittavan

sitä, että jaettaessa kaksi lukua keskenään jako menee tasan, saman asian voi selvittää ilman ohjelmointiakin laskukoneella.

Opettaja voi päättää teetättää oppilailla myös molemmat versiot. Jos ensin tehdään helpompi, oppilaat ehkä huomaavat itse ohjelman puutteet (käyttäjän on itse manuaalisesti syötettävä kaikki testattavat luvut), ja oivallus monipuolisemman eli sen haastavamman version tarpeesta tulee ehkä oppilailta. Vaihtoehtoisesti opettaja voi heti alkuun esitellä valmiina helpomman version, ja kysyä oppilailta, miten ohjelmaa pitäisi muuttaa, jos haluttaisiin selvittää nopeasti kaikki luvut, joilla jokin luku on jaollinen.

Haastavampi versio tutustuttaa oppilaat siihen, että kone kykenee tekemään ihmistä nopeammin toistoa vaativia tehtäviä. **Oppilaita voi motivoida esimerkiksi niin, että opettaja tuo luokkaan laatikon, jossa on numerokoodilukko. Oppilaille voidaan kertoa, että laatikossa on jotain, mitä he haluavat, ja lukon avauskoodi on eräs nelinumeroinen luku, jolla luku 13265 on jaollinen. Tehtävään voidaan tuoda haastetta sanomalla, että oppilailla on vain kolme yritystä, ja mikäli lukko ei aukea kolmannella, sen sisältö tuhoutuu.** Hieman pakohuonemaista tunnelmaa siis. Koska yrityksiä on vain kolme, oppilaat eivät lähde arvailemaan lukon koodia.

Oppilaat toivottavasti ymmärtävät, että nelinumeroisia lukuja, joilla jaollisuutta tulisi kokeilla on tuhansia. Yksi kerrallaan kokeilemalla ratkaiseminen tai esimerkiksi opettajan taululla esittelemällä helpommalla ohjelmalla kokeileminen on siksi työlästä ja hidasta. **Tästä päästään oppilaiden kanssa keskusteluun, mitä ohjelman pitäisi tehdä eri tavalla, jotta ratkaisun etsiminen helpottuisi.** Oppilaiden ideat ohjelman vaatimuksista kannattaa kerätä taululle.

Ohjelman tulisi yhdellä käyttäjän syötteellä tutkia ja kertoa kaikki luvut, joilla syötetty luku on jaollinen. Juuri näin tämän tuntisuunnitelman vaikeampi ohjelma tekee, ja sen koodaamalla oppilaat saavat hetkessä selville, että nelinumeroisia lukuja, joilla tuo luku 13265 on jaollinen, on vain 2.

Ohjelma **kysyy käyttäjältä lukua (mitä tahansa lukua), selvittää sitten millä kaikilla luvuilla välillä ykkösestä kysyttyyn lukuun kysytty luku on jaollinen.** Ohjelma listaa kaikki luvut, joilla kysytty luku on jaollinen listaan ja näyttää nämä lopuksi käyttäjälle. Periaatteessa tämä vaikeampi sovellus selvittäisi minkä tahansa luvun jaollisuuden, mutta koska Scratchissa listojen suuruus on rajattu 200000:n, luvun on oltava väliltä 0-200000. Tarkemmat ohjeet vaikeampaan tehtävään jäljempänä.

Eriyttäminen

Helpommassa versiossa on myös ylös- ja alaspäin eriyttävät versiot, jotta opettaja voi soveltaa suunnitelmaa käytettävän ajan ja taitotason mukaisesti. Vaikeampaa versiota voi eriyttää esimerkiksi niin, että lähettää oppilaille linkillä puolivalmiin koodin, jota oppilaiden täytyy muokata tai niin, että lähettää oppilaille projektin, jossa kaikki tarvittavat palat ovat jo työpöydällä, oppilaiden täytyy vaan järjestellä ne oikeaan järjestykseen.

Tehtävä 1 (helpompi versio)

Käytettävät/harjoiteltavat ohjelmointitaidot:

- syötteen pyytäminen käyttäjältä ja sen tallentaminen muuttujaan

- matemaattiset operaatiot (jakolasku ja vertailu)
- ehtolause (jos-muuten)
- silmukka (toista kunnes)

Oppilaat työskentelevät pareittain.

Tehtävänanto (suluissa opettajalle tarkoitetut tarkennukset)

Suunnitelkaa parin kanssa ohjelma, jossa valitsemanne Scratch-hahmo kysyy ensin käyttäjältä kaksi lukua ja laskee sitten, onko ensimmäinen luku jaollinen toisella luvulla. Tämän jälkeen ohjelman tulee ilmoittaa käyttäjälle, onko jaettava jaollinen jakajalla. (*perusversio)*

Jaollisuuden ilmoittamisen jälkeen ohjelman tulisi kysyä käyttäjältä, haluaako tämä tutkia toisia lukuja. Mikäli käyttäjä vastaa myöntävästi, ohjelman tulisi toistaa sama koodi uudelleen. Mikäli käyttäjä vastaa kielteisesti, ohjelman pitäisi loppua. *(Tämä on hieman enemmän taitoa vaativa sovellus. Tässä tarvitaan toista kunnes -luuppia.)*

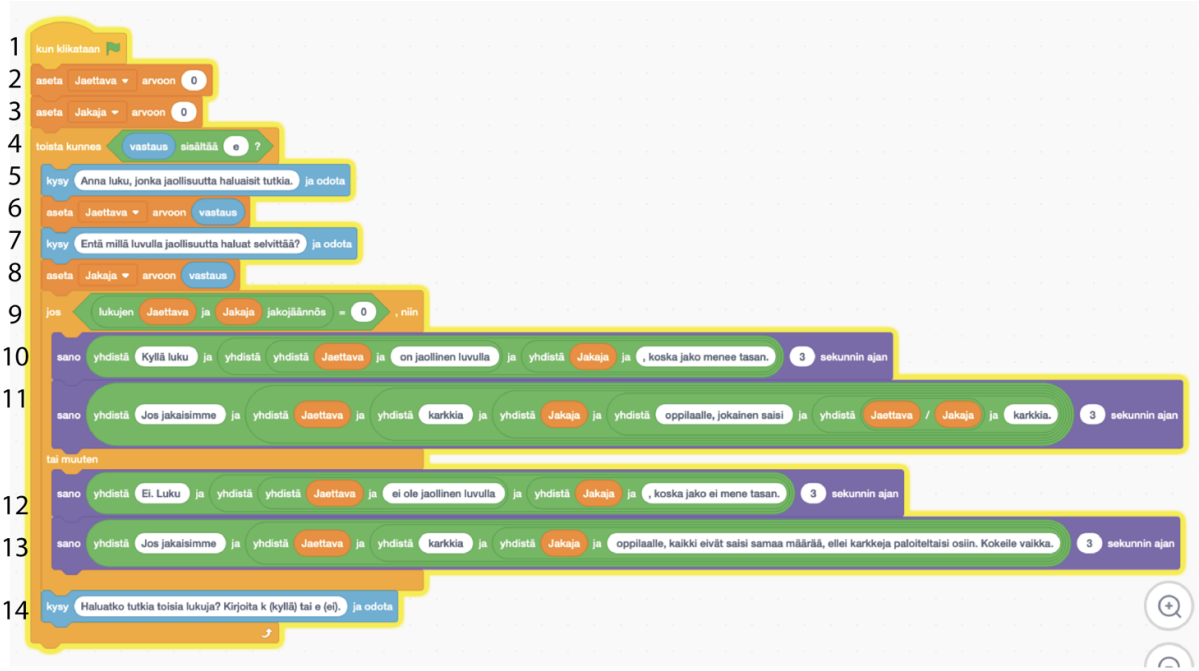
Mikäli olette nopeita ja tehtävä on teille helppo, ohjelmoikaa ohjelma perustelemaan käyttäjälle esimerkin avulla, miksi jaettava ei ole jaollinen jakajalla. *(Ohjelma voi vaikkapa antaa esimerkin, jossa kyseisillä luvuilla jako ei mene tasan.)*

Ongelman pilkkominen osiin

Aloittakaa pohtimalla pareittain tai koko luokan kesken kynällä ja paperilla, mitä osatehtäviä ohjelman täytyy osata tehdä. Voitte myös toimia niin, että yksi oppilaista tai opettaja toimii koneena.

- Ohjelman täytyy kysyä käyttäjältä kaksi lukua ja ottaa nämä talteen johonkin. Tässä muuttujat ovat tarpeen. Tätä voi havainnollistaa niin, että ope kysyy käyttäjältä kaksi lukua ja kirjoittaa ne taululle.
- Ohjelman täytyy seuraavaksi laskea, meneekö kahden luvun jako tasan, eli jääkö jakojäännöstä.
- Ohjelman täytyy ilmoittaa jotain käyttäjälle. Jos jako jäännös on 0, ohjelma ilmoittaa eri asian kuin jos jakojäännös ei ole 0.

Esimerkkiratkaisun löydät tämän linkin takaa. Alla koodi kuvana selityksineen.



Rivi 1: lippu käynnistää ohjelman

Rivit 2-3: Muuttujat osiossa on luotu valmiiksi muuttujat Jaettava ja Jakaja, jotka ohjelman käynnistyessä nollataan.

Rivit 4-14: Ohjelma toistaa näiden välissä olevia asioita järjestyksesä niin kauan, kunnes käyttäjä vastaa rivillä 14 olevaan kysymykseen e.

Rivi 5: Ohjelma pyytää käyttäjältä syötettä.

Rivi 6: Ohjelma sijoittaa pyytämänsä syötteen talteen muuttujaan Jaettava.

Rivit 7-8: Ohjelma toistaa saman, nyt muuttujalle Jakaja.

Rivit 9-13: Ohjelma laskee rivillä 9 Jaettavan ja Jakajan osamäärän ja jos jakojäännös on 0 (jako menee tasan, eli Jaettava on jaollinen Jakajalla), ohjelma tekee riville 10-11 ohjelmoidut toimenpiteet. Huomaa, kuinka muuttujat on upotettu virkkeisiin yhdistelemällä vihreitä palikoita.

Jos taas jako ei mene tasan, eli jakojäännös ei ole 0, ohjelma hyppää riville 12 ja jatkaa siitä riville 13.

Suoritettuaan jakojäännöksestä riippuen rivit 10-11 tai 12-13, ohjelma kysyy käyttäjältä rivillä 14 syötettä ja hyppää taas riville 4 tai lopettaa ohjelman käyttäjän syötteestä riippuen.

Tehtävä 2 (haastavampi versio)

Käytännössä tehtävän 1 ohjelma vain laskee jakolaskun ja ilmoittaa sitten menikö jako tasan. Saman voisi siis selvittää pelkkää laskinta käyttäen. Haastavammassa versiossa käyttäjä voi syöttää minkä tahansa luvun ja ohjelman selvittää kaikki luvut, joilla tuo luku on jaollinen.

Käytettävät/harjoiteltavat ohjelmointitaidot:

- syötteen pyytäminen käyttäjältä ja sen tallentaminen muuttujaan
- listojen käyttö (listoja tarvitaan, jotta ohjelma voi käydä järjestyksessä läpi jokaisen luvun ja listata mitkä luvuista jakavat kysytyn luvun tasan
- matemaattiset operaatiot (jakolasku ja vertailu)
- ehtolause (jos-muuten)
- silmukka (toista kunnes)

Oppilaat työskentelevät pareittain.

Tehtävänanto (suluissa opettajalle tarkoitetut tarkennukset)

Tuo luokkaan laatikko, jossa on numerokoodilukko. Laatikossa voi olla sisällä jotain, mitä oppilaat haluavat. Mikäli lukkoa ei ole saatavilla, voit myös ohjelmoida itse miniohjelman ja kertoa, että syöttämällä ohjelmaan oikean nelinumeroisen koodin, oppilaat saavat tiedon aarteen sijainnista (aarre voi olla jotain, mitä oppilaat haluavat). Vaihtoehtoisesti tämä voi olla vaikkapa Google Forms -kysely tai vastaava, jossa oikea vastaus tekstikenttäkysymykseen avaa oppilaille sivun, jossa viesti on.

Kerro oppilaille seuraava.

”Lukon avauskoodi on eräs nelinumeroinen luku, jolla luku 13265 on jaollinen. Ratkaisun löytäminen on siinä mielessä haastavaa, että teillä on vain kolme yritystä, ja mikäli lukko ei aukea kolmannella, sen sisältö tuhoutuu. ” (Oppilaille voi tarvittaessa myös kertoa, että on olemassa vain kaksi nelinumeroista lukua, jolla tuo luku on jaollinen. Oppilaat saavat siis tehdä yhden virheenkin koodin syöttämisessä.)

Pohtikaa tämän jälkeen yhdessä, miksi aiemmalla tunnilla tehty (tai opettajan taululla esittelemä) ohjelma ei ole hyvä tehtävän ratkaisussa. (Siinä jokaista lukua on kokeiltava manuaalisesti, eli oppilaiden pitäisi lähteä luvusta 1000 ja edetä yksi kerrallaan lukuun 10000 asti.)

Pohtikaa, mitä ohjelman tulisi tehdä, jotta se olisi tehokas ja kerätkää ohjelman vaatimukset yhdessä taululle (ongelman pilkkominen osiin)

Opettajalle tiedoksi:

- Ohjelman täytyy kysyä käyttäjältä lukua ja ottaa tämä talteen johonkin (**muuttuja**).
- Ohjelman täytyy aloittaa ykkösestä ja testata kaikilla luvuilla, meneekö jako tasan. (Tämän voisi ratkaista ainakin **listoja** käyttäen. Ohjelmaan voisi esimerkiksi luoda kaksi listaa, joista ensimmäiseen ohjelma laittaa järjestyksessä kaikki luvut ykkösestä (tai nollasta) annettuun lukuun saakka ja alkaa sitten testata listan ensimmäisestä luvusta alkaen, meneekö jako tällä luvulla tasan. Jos jako menee tasan, ohjelma laittaa tämän luvun toiseen listaan talteen, poistaa sen sitten ensimmäisestä listasta ja jatkaa testaamaan seuraavaa lukua. Jos jako ei mene tasan, ohjelma vain poistaa luvun listasta ja siirtyy seuraavaan lukuun. Ohjelma toistaa tämän luupin niin monta kertaa, kuin listalla on lukuja.)
- Ohjelman täytyy kerätä talteen luvut, joilla luku on jaollinen.
- Ohjelman täytyy lopuksi ilmoittaa, millä luvuilla kysytty luku oli jaollinen

Kerro taas oppilaille seuraava tehtävänanto.

”Suunnitelkaa ja ohjelmoikaa parin kanssa (tai ryhmänä) ohjelma, joka tekee listaamanne asiat. Valitsemanne Scratch-hahmo kysyy ensin käyttäjältä luvun ja selvittää sitten kaikki luvut, joilla tuo luku on jaollinen. Ohjelman tulisi ilmoittaa käyttäjälle jollain tapaa, mieluiten luettelona tai listana, millä luvuilla luku on jaollinen.

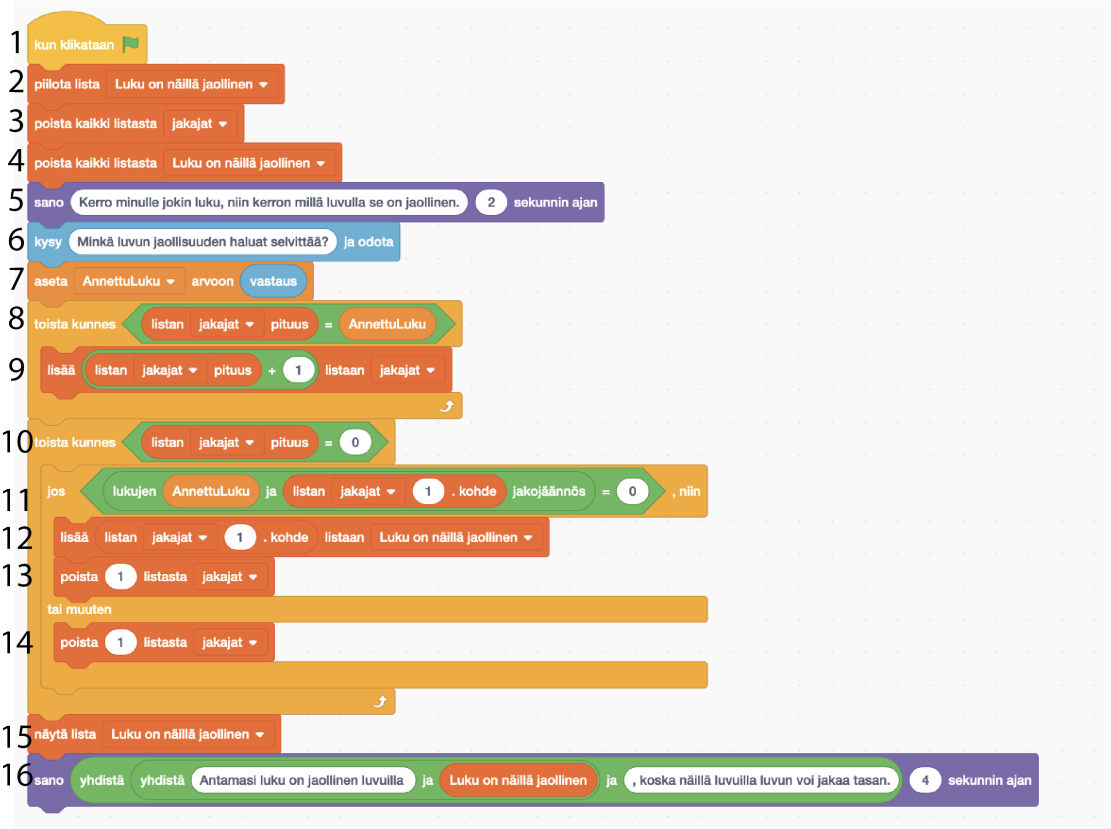
Mikäli olette nopeita ja tehtävä on teille helppo, täydentäkää ohjelmaa esimerkiksi niin, että…

- Ohjelma kertoo, onko kyseessä parillinen luku.
- Ohjelma kertoo, jos kyseessä on alkuluku.

Ohjaaminen tehtävän aikana

- Ohjaa oppilaita testaamaan ohjelmaa osissa:
  - Toimiiko käyttäjän syötteen kysyminen?
  - Osaako ohjelma luetella listaan luvut ykkösestä annettuun lukuun saakka?
  - Osaako ohjelma ottaa ensimmäisen luvun listalta ja testata sitä?
  - jne jne
- Pyydä oppilaita puhumaan ongelmatilanteissa ääneen tai piirtämään paperille, mitä ohjelma tekee ja mitä sen pitäisi tehdä.

Täältä löydät esimerkkiratkaisuni, ja alla kommentoitu kuva koodista.



Rivi 1: Ohjelma käynnistyy vihreällä lipulla.

Rivit 2-4: Ohjelmalle on luotu listat ”Luku on näillä jaollinen sekä ”jakajat”. Lista jakajat on oletuksena piilotettu, mutta lista ”Luku on näillä jaollinen” tulee näkyviin aina ohjelman lopussa. Sen vuoksi molemmat listat täytyy tyhjentää ja lista ”Luku on näillä jaollinen” täytyy alussa piilottaa.

Rivi 5: Ohjelma antaa käyttäjälle ohjeen.

Rivi 6: Ohjelma kysyy käyttäjältä syötettä ja tallentaa käyttäjän syötteen muuttujan ”AnnettuLuku” arvoksi. Muuttuja täytyy luoda muuttujissa.

Rivit 8-9: Ohjelma lisää lukuja listaan ”jakajat” yksi kerrallaan aloittaen luvusta 0+1 (koska listan jakajat pituus on alussa 0, eli lista on tyhjä) ja lisäten aina yhden edelliseen numeroon. Ohjelma toistaa rivin 9 niin monta kertaa, että listan pituus on yhtä suuri kuin annettu luku. Toisin sanoen ohjelma luettelee listaan järjestyksessä kaikki luvut luvusta 1 lukuun, jonka käyttäjä syötti syötteenä.

Rivit 10-14: Tämän jälkeen ohjelma alkaa toistaa riveillä 11-14 ohjelmoituja asioita, kunnes lista jakajajat on tyhjä eli käyty luku luvulta läpi. Ensin ohjelma testaa, onko annettu luku listan ensimmäisellä luvulla (1) jaollinen. Jos on, ohjelma siirtää tämän luvun toiselle listalle, nimeltä ”Luku on näillä jaollinen” (rivillä 12), poistaa luvun listalta ”jakajat” (rivillä 13) ja siirtyy uudelleen riville 11. Nyt listan ”jakajat” ensimmäinen luku on luku 2, koska luku 1 siirrettiin ja poistettiin tältä listalta. Ohjelma toistaa taas rivin 11. Jos annettu luku ei olekaan jaollinen tällä luvulla, ohjelma hyppää suoraan riville 14, eli poistaa luvun listasta ”jakajat” siirtämättä sitä toiselle listalle, ja aloitata tämän jälkeen taas riviltä 11, nyt luvulla 3, joka on nyt listan ”jakajat” ensimmäinen luku. Ohjelma toistaa tämän uudelleen ja uudelleen, kunnes listalla ”jakajat” ei ole enää lukuja (Huomaa rivi 10: Toista kunnes listan jakajat pituus = 0).

Rivi 15-16: Kun lista jakajat on käyty loppuun, ohjelma hyppää riville 15, eli näyttää valmiin listan ”Luku on näillä jaollinen” ja hahmo sanoo vielä rivillä 16 millä luvuilla luku oli jaollinen.

Ohjelman liittäminen laajempaan kontekstiin ja jaollisuuden tutkiminen ohjelman avulla

Pohdintatehtäviä

1. Minkä ongelman tehtävän koodilukkotehtävässä tietokoneohjelma ratkaisi? Miksi ohjelma oli hyödyllinen?
2. Miksi ongelma oli helppo ratkaista ohjelmalla? (Ratkaisun löytäminen edellytti monien erilaisten vaihtoehtojen läpikäymistä eli saman tehtävän toistoa, minkä tietokone tekee nopeasti.)
3. Pohtikaa yhdessä vastaavia arkipäivän ongelmia, joissa samalla periaatteella toimiva, listaa toistoluupin avulla läpikäyvä ohjelma auttaa.
4. Tutkikaa ohjelman avulla jaollisuutta:
    a. Riippuuko jakajien määrä luvun suuruudesta, eli onko totta, että mitä suurempi luku on, sitä useammalla luvulla se on jaollinen?

Algoritmista ajattelua liikkuen ja ilman tietokoneita, 4.lk

11.6.2020

Tuntisuunnitelma 4.lk

Opetus toteutettaisiin matematiikan ja liikunnan tunneilla.

Tavoitteet: Oppilas tutustuu algoritmiseen ajatteluun, ymmärtää missä kaikkialla sitä voi hyödyntää ja oppii yksinkertaista ohjelmointia.

Tunti 1:

Valmistelut: Opettaja hakee oppilaille henkilökohtaisen laitteen sekä as

Tutustutaan algoritmiseen ajatteluun.

- keskustellaan ennakko-oletuksista:
  - mitä ohjelmointi on?
  - missä sitä käytetään?
- kokeillaan yksinkertaista ohjelmointia
  - opettaja antaa yksinkertaisia käskyjä, joiden mukaisesti oppilaat toimivat (ks. Ohje1)
  - harjoitellaan koodaamista code.org harjoitusten avulla.

Tunti 2:

Esterata ulkona.

Valmistelut:

Opettaja tekee radan pihalle esimerkiksi puistoalueelle. Opettaja sitoo narua puiden ympärille luoden polun tai pienen labyrintin. Radalla voi olla myös pieniä esteitä, joiden takia oppilaan pitää kumartua.

- Oppilas näkee esteradan ulkona, kävelee sen läpi.
- Oppilas valitsee valmiista käskyistä sopivat ja muodostaa niistä koodin.
- Toinen oppilas suorittaa radan koodin mukaisesti. Koodirivin tehnyt oppilas tarkkailee, miten koodi toimii ja tarvittaessa muokkaa koodiriviä seuraavalle kokeilukierrokselle.
- Esteradan saa yrittää läpäistä kaksi kertaa.
- Oppilaat odottavat esteratavuoroaan luokassa ohjelmoiden https://code.org harjoitusten avulla.

Arviointi:

- opettaja arvioi tunnin 1 ja 2 harjoituksissa suoriutumista
  - onnistuuko ohjeiden mukainen toiminta
  - pystyykö muodostamaan koodin esteradan suorittamiseksi
- vertaisarviointi esterataa suorittaessa

        - mitä tulisi muuttaa
        - toimiiko koodi
    - itsearviointi ja reflektointi
        - onnistuiko koodi, tekisitkö jotain toisin

Tarvittavat materiaalit:

paperia, kyniä, mobiililaite/tietokone, käskykortit, välineitä esteradan luomiseksi (esim. narua ja tötsöjä)

Materiaali1: Sopivia käskyjä (tulostettu valmiiksi):

Askel eteenpäin

Käänny oikealle

Käänny vasemmalle

Askel taaksepäin

Kyyristy

Nouse

Määräkortteja x1, x2, x3, x4, x5, x6, x7, x8, x9, x10

Ohje1:

1. Hae A4-paperi pöydältä
2. Laita paperi pulpetille
3. Ota kynä kirjoituskäteen
4. Aseta kynän terä paperin vasempaan ylälaitaan
5. Piirrä suora viiva alaspäin
6. Pysähdy paperin vasemmassa alalaidassa
7. Piirrä suora viiva oikealle päin
8. Pysähdy paperin oikeassa alalaidassa
9. Piirrä suora viiva ylöspäin
10. Pysähdy paperin oikeassa ylälaidassa
11. Piirrä suora viiva vasemmalle päin
12. Pysähdy vasemmassa ylälaidassa

Mikä kuvio syntyi?

BeeBot-ohjelmointitehtävän suunnittelu, 4.lk

25.10.2020

BeeBotin ohjelmointitehtävän suunnittelu 1.-2.-luokkalaisille (4.lk)

Aihepiirin valinta ja rajaus

Aihe

Projektin aiheena on ohjelmointitehtävän suunnittelu 1.-2.-luokkalaisille. Projekti on tarkoitettu neljäsluokkalaisille tai sitä vanhemmille. Ohjelmointitehtävän suunnittelu tehdään parin kanssa. Ohjelmointitehtävässä käytetään BeeBot-robotteja ja tehtävän aihe liittyy 1.-2. luokan matematiikan tavoitteisiin.

Ennakkotiedot

Neljäsluokkalaiset oppilaat ovat aiemmin käyttäneet BeeBot-robotteja omassa ohjelmointityöskentelyssään, joten niiden toimintaperiaatteet ja peruskäsitteet ovat heille tuttuja. Lisäksi alkuopetuksen matematiikan aiheet ovat heille tuttuja ja helppoja. Myös *1.-2.-luokkalaiset* ovat tutustuneet BeeBot-robotin toimintalogiikkaan, joten sen opastamiseen ei mene aikaa. *Opettajan* tulee tuntea BeeBot-robotin toimintalogiikka ja alkuopetuksen matematiikan aiheet.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Tavoitteet

Projektin tarkoituksena on luoda BeeBot-robotin ohjelmointitehtäviä pienemmille oppilaille. Tehtävien tulee liittyä matematiikan aiheisiin. *Neljäsluokkalaisten osalta* projektin tavoitteena on kehittää oppilaiden ohjelmointikykyä ja algoritmista ajattelua, harjoitella matematiikan aiheeseen liittyvän ohjelmointitehtävän suunnittelua ja tuottamista sekä virheenjäljityskykyä. Tavoitteena on myös kehittää oppilaiden kehittää oppilaiden ongelmanratkaisu- ja yhteistyötaitoja.

- S1 Ajattelun taidot - Suunnitellaan ja toteutetaan ohjelmia graafisessa ohjelmointiympäristössä.
- T1 - Pitää yllä oppilaan innostusta ja kiinnostusta matematiikkaa kohtaan sekä tukea myönteistä minäkuvaa ja itseluottamusta
- T5 - Ohjata ja tukea oppilasta ongelmanratkaisutaitojen kehittämisessä
- T14 - Innostaa oppilasta laatimaan toimintaohjeita tietokoneohjelmina graafisessa ohjelmointiympäristössä

1.-2. luokan oppilaiden osalta tavoitteena on BeeBot-robottiin liittyvien ohjelmointitaitojen ja toimintalogiikan vahvistuminen, algoritmisen ajattelun kehittyminen, matematiikan aiheen kertaus sekä ongelmanratkaisu- ja yhteistyötaitojen kehittyminen.

- S1 Ajattelun taidot - Tutustuminen ohjelmoinnin alkeisiin alkaa laatimalla vaiheittaisia toimintaohjeita, joita myös testataan.
- T1 - Tukea oppilaan innostusta ja kiinnostusta matematiikkaa kohtaan sekä myönteisen minäkuvan ja itseluottamuksen kehittymistä
- T4 - Ohjata oppilasta kehittämään päättely- ja ongelmanratkaisutaitojaan
- T12 - Harjaannuttaa oppilasta laatimaan vaiheittaisia toimintaohjeita ja toimimaan ohjeen mukaan

Arviointi

Opettaja havainnoi ja arvioi oppilasparien toimintaa ja osallistumista projektin aikana. Hän antaa oppilaille ohjaavaa palautetta. Opettaja arvioi myös projektin tuotoksen toimivuutta eli BeeBot-robotin ohjelmointitehtävää.

Neljäsluokkalaiset oppilaat antavat palautetta osallistumisestaan toimintaan ja tuotoksestaan tekemällä itse- ja vertaisarvioinnin. He arvioivat mm. ohjelmointitehtävän suunnittelua, toteutusta ja toimivuutta, työskentelyä parin kanssa, avun tarvetta ja ohjausta sekä kertovat mielipiteensä tehtävästän innostavuudesta ja haasteellisuudesta.

1.-2.-luokkalaiset ja heidän opettajansa antavat käyttäjäpalautetta tehtävän toimivuudesta. Lisäksi 1.-2.-luokkalaisten opettaja arvioi tehtävän sopivuutta ko. matemaattisen aiheen kertaukseen.

Työskentelyvälineet ja opetusmenetelmät ja ympäristö

Välineet

Ohjelmointitehtävän ohjeistus kirjataan Slides-ohjelmalla ja se jaetaan luokan yhteiseen Classroomiin. Lisäksi työskentelyssä tarvitaan BeeBot-robotit ja niiden alustat. Alustoina käytetään läpinäkyviä alustoja, joille kiinnitetään oppilaiden suunnittelemat ja valmistamat kortit. Itse- ja vertaisarvioinnit kirjoitetaan valmiiseen arviointipohjaan, joka löytyy Classroomista.

Motivointi

Oppilaita motivoidaan projektiin antamalla heille ensin aikaa kerrata BeeBot-robottien käyttöä ja toimintaohjeita. Heidän tulee tietää, miten BeeBot käynnistetään, miten BeeBotia liikutetaan, miten BeeBotille syötetyt käskyt nollataan, miten äänitoimintoa käytetään jne. Oppilaat saavat tehdä toisilleen suullisesti pieniä ohjelmointitehtäviä kaupunki- ja saarialustoille. Projektissa motivoivaa on myös pelin tekeminen pienemmille oppilaille, jotka tulevat testaamaan ja arvioimaan pelin toimivuutta.

Ohjeistus

- Tarkoituksena on suunnitella 1.-2.-luokkalaisten käyttöön BeeBot-roboteilla suoritettava ohjelmointitehtävä.
- Tehtävän tulee liittyä matematiikkaan ja aihealueina ovat lukujen 4-9 hajotelmat, kymppiparit, kymmenylitys lukualueella 0-20, kertotaulut 2-5 ja 10, tasokuviot ja kellonajat.
- Ohjelmointitehtävä tehdään yhdessä parin kanssa.
- Tehtävässä tulee olla vähintään kymmenen kohtaa.
- Tehtäviä voi vaikeuttaa esimerkiksi asettamalla esteitä robottien kulkureitille tai ohjeistamalla robotteja aina palaamaan takaisin lähtöpisteesen.
- Ohjelmointitehtävän suunnitelma ja ohjeet kirjoitetaan Slides-ohjelmalla, Ohjelmointikurssille Classroomiin. (liite)
- Valmista tehtävää testataan 1.-2.-luokkalaisilla ja he saavat antaa tehtävän toimivuudesta palautetta. Tehtävää voi myös esitestata luokkakavereilla.
- Lopuksi annetaan itselle ja parille arvio projektityöskentelystä ja tuotoksen toimivuudesta.

Projektin aikataulu

1. oppitunti
   - Projektin tavoitteiden esittely
   - BeeBoti-robottien käytön ja toimintaohjeiden kertaus
   - Projektin ohjeistuksen läpikäyminen
   - Projektityöskentelyn aloitus

2. oppitunti
- Jatketaan projektityöskentelyä
- Opettaja seuraa työskentelyä ja antaa ohjaavaa palautetta
- Itse- ja vertaisarviointi

3. oppitunti
- 1.-2.-luokkalaisten ohjeistus
- Ohjelmointitehtävän testaus, jolloin isommat oppilaat toimivat opettajan apuna esim. ohjeiden lukemisessa.
- Käyttäjäpalaute 1.-2.-luokkalaisilta ja opettajalta

Lähteet ja ohjeistukseen käytettävät materiaalit

Video BeeBotin käytöstä
Ideoita BeeBotin käyttöä varten
Ideoita ohjelmointimattoon
BeeBot ohjelmointikortit
Ohjelmointikortit

Linkkejä vapaasti käytettäviin kuviin:
Creative commons
Papunet
Pixabay

Liitteet

Ohjelmointitehtävän suunnitelma (4.lk)

Alkuohjeistus 1.-2.-luokkalaisille

- Mitä matematiikan aihealuetta tehtävät käsittelevät ja mille luokka-asteelle tehtävät on tarkoitettu?
- Mitä alkuvalmisteluja tehtävä vaatii?
- Miten BeeBot-robotti toimii? (lyhyt ohjeistus)

Tehtäväohjeistukset 1.-2.-luokkalaisille

- Vähintään kymmenen tehtävää
- Kirjoittakaa selkeä ohje jokaiseen tehtävään.
- Esimerkkitehtävä: Ratkaise kymppiparit
    - Kuljeta BeeBot-robotti oikeaan kohtaan. Käännä kortti ja kirjoita laskun perään kortin takana oleva kirjain.
    - Mikä luku tulee viivalle? 4 + ___ = 10
- Kirjoita suunnitelmaan myös oikeiden vastausten koodit.
- Jos tehtävään tarvitaan esim. vastauspaperi, tehkää siihen valmis pohja.

Koordinaatisto ja ohjelmointi, 4.lk

Jonna Hakala, CC BY-SA 4.0

18.9.2020

AIHEPIIRIN VALINTA JA RAJAUS

Haluan käyttää ohjelmointia matematiikassa koordinaattien opettamisen yhteydessä 4. luokalla, koska ohjelmointi sopii syventämään koordinaatiston sisäistämistä, kun ohjelmointi tapahtuu koordinaatistossa. Opetuksen tavoitteena on, että oppilas pääsee itse tutkimaan koordinaatistoa pelillisyyden kautta.

1. tunti: Tarkoituksena on ensin liikkua koordinaatistossa hedelmiä keräten. Hedelmät on asetettu oppilaille tulostettuun ruudulliseen koordinaatistoon. Oppilaan on suunniteltava Petterille reitti hedelmien keräämiseen, kun Petteri lähtee origosta (keskeltä) liikkeelle. Tämän jälkeen oppilaille esitellään tarkemmin koordinaatistoa matemaattisin termein. Tämän jälkeen oppilaille kerrotaan, että Petteri osaakin puhua vain koordinaatistokieltä. Ohjeet Petterille onkin muutettava siis matemaattisille termeille koordinaatistosta, esim. ”liiku x-akselilla positiiviseen suuntaan kahden ruudun verran”.

2. tunti: Oppilaille kehitellään lattialle koordinaatisto, jossa he pystyvät itse liikkumaan ohjelmointileikissä. Oppilaat pareineen antavat toisilleen vuorotellen ohjelmapatteriston liikkua koordinaatistossa, esim. saavuttaakseen jonkun esineen tai pisteen. Opettaja ohjaa oppilaita käskyttämään paria matemaattisin termein. Tämän tunnin loppuun opettaja teettää oppilailla perinteisempiä tehtäviä koordinaatistosta ja arvioi niiden onnistumisen perusteella oppimisen tuloksia sekä kiinnittää huomiota oppilaiden itsearviointiin.

VALITTUUN AIHEPIIRIIN LIITTYVIEN OPPIMISTAVOITTEIDEN MÄÄRITTELY JA ARVIOINTI

Oppimistavoitteet:

1. Oppilas osaa sekä käyttää että liikkua koordinaatistossa sujuvasti

2. Oppilas tietää koordinaatiston suunnat sekä osaa tulkita koordinaatistosta vaadittuja tietoja

3. Oppilas osaa piirtää itse koordinaatiston

4. Oppilas osaa noudattaa annettuja ohjeita

5. Oppilas ymmärtää ohjelmoinnin perusperiaatteen: ohjelmointiohjeita noudattamalla saavutetaan haluttu tavoite

Opettajan arviointi: Opettaja käyttää tilanteessa havainnointia arvioidakseen oppilaiden ymmärrystä tehtävistä niiden sujuvuuden ja komentojen kielentämisen kautta. Lisäksi opettaja arvioi 2. tunnin lopussa olevien tehtävien perusteella oppilaiden osaamista.

Oppilaan itsearviointi: Oppilaille voidaan esittää omaa toimintaa pohdituttavia kysymyksiä pitkin prosessia. Lisäksi 2. tunnin lopussa annettavan tehtäväpaketin loppuun lisätään jokin itsearvioinnin elementti, esim. hymiöt hyvästä sujumisesta (hymynaama) huonoon (surunaama).

TYÖSKENTELYVÄLINEET JA OPETUSMENETELMÄT

Välineet: opettajan tulostamat hedelmäkeräyspelit, teippiä

Ohjeistus: Opettaja antaa oppilaille lyhyet ohjeet tehtäviin, mutta tarkoituksena on, että oppilas saa ensin itse selvittää koordinaatiston toimivuutta tutkimuksellisen oppimisen periaattein. Opettajan suurin ohjeistuksen anto liittyy koordinaatiston tarkempaan opettamiseen matemaattisin termein, jolloin oppilaiden kanssa käydään keskustelua siitä, miten äsken Petterille annetut ohjeet annettaisiinkin nyt koordinaatistokielellä.

Motivointi: Toiminnan pelillinen, tutkiva ja yhteisöllinen luonne toimii motivoijana.

Yhteistyö: Yhteistyö sujuu toivon mukaan luontevasti pelillisen luonteen takia.

Aiemman osaamisen huomiointi: Oppilaat saavat vapaasti ilmaista aiempia kokemuksiaan ohjelmoinnista keskusteluissa sekä mahdollisesti auttaa muita.

Opitun yhdistäminen: Uuden tiedon kasautuessa vanhan päälle, voidaan leikkeihin ja peleihin lisätä elementtejä uusista asioista.

Algoritmeja toiminnallisesti, 4.lk

Anitta Leipälä, CC BY-SA 4.0

17.11.2020

ALGORITMEJA TOIMINNALLISESTI 4. lk, 9 oppilasta

Tuntisuunnitelma-työpaja Tekijä Anitta Leipälä

Aihepiirin valinta ja rajaus

Sain uuden erityisluokan lukukauden alussa. Oppilaat eivät ole aiemmin tottuneet toiminnalliseen matematiikkaan eikä systemaattiseen ongelmanratkaisuun. Koska sanalliset tehtävät ja kaikenlainen päättely ovat vaikeita, halusin harjoitella algoritmeja toiminnallisesti ennen varsinaisen ohjelmoinnin aloittamisen pohjaharjoituksina. Oppilailla on kielellisiä vaikeuksia ja erityisesti suomen kielen suullinen ja kirjallinen tuottaminen ovat haastavia. Tästä syystä halusin ottaa mukaan tehtäviä, joissa tarvitaan matematiikkapuhetta, mutta pitää myös tuottaa selkeitä kirjallisia ohjeita. Aiempia ennakkotietoja tai taitoja ei tarvita.

Kahden-kolmen oppitunnin aikana oppilaat tutustuvat algoritmiseen ajatteluun, sekä oppivat ohjelmoinnin peruskäsitteistä lause/käsky/muuttuja.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Ensimmäinen (ja toinen) tunti (1-2h)

Tunnin tavoitteet

- Oppilas tutustuu ohjelmoinnillisen ajattelun perusteisiin.
- Oppilas kehittää omaa algoritmista ajatteluaan ja yhteistyötaitojaan toimiessaan yhteistyössä muiden oppilaiden kanssa.
- Oppilas harjoittelee kirjoittamaan ja muokkaamaan algoritmia.

Aiheeseen virittäytyminen

Pohditaan sitä, mistä osatoiminnoista erilaiset toiminnat koostuvat. Oppilaat saavat keksiä erilaisia toimintoja, ja pohtia, mitä eri vaiheita niissä on.

Hanoin torni

Toimintoja harjoitellaan ongelmanratkaisua ja loogista päättelyä vaativalla tehtävällä. Aluksi ratkaistaan ongelmanratkaisutehtävä ”Hanoin torni” (https://www.innokas.fi/wp-content/uploads/2018/02/La%CC%88hto%CC%88laukauskoodaukseen.pdf) pareittain.

Opettaja arpoo parit ja jokaiselle parille seuraavat tarvikkeet:

- kolmeen osaan jaettu alusta
- kolme erikokoista kappaletta, jotka voidaan asettaa päällekkäin

Opettaja näyttää tehtävän kuvan ja ohjeistaa käyttämään paljon puhetta tehtävän ratkaisemiseksi. Parit ryhtyvät työhön. Kun tehtävä on ratkaistu, parit kirjoittavat ratkaisuohjeet paperille vaiheittain. Kun kirjalliset ohjeet ovat valmiit, ne testataan vielä tietokoneella Tower Hanoi tehtävässä:

http://www.dynamicdrive.com/dynamicindex12/towerhanoi.htm

Kotiläksyksi tulee teettää sama tehtävä kotona jollekin perheenjäsenelle.

Tuntien lopuksi katsotaan vielä yhdessä video (https://www.youtube.com/watch?v=j-6N3bLgYyQ) voileivän valmistamisesta. Opettaja pysäyttää videon alussa jossain sopivassa kohdassa varmistaen, että oppilaat ymmärtävät, mistä siinä on kyse. Videolla isä noudattaa kirjaimellisesti lastensa kirjoittamia vaiheittaisia ohjeita voileivän valmistamisesta.

Videon katsomisen jälkeen keskustellaan yhdessä videon herättämistä ajatuksista. Tämän jälkeen annetaan tehtävä seuraavaa tuntia varten:

> Keksikää jokin tehtävä, jonka voi suorittaa omassa luokassa. Mieti, miten ohjeistaisit sen vaiheittain niin, että toinen osaa toimia oikein antamasi ohjeen mukaisesti.

Kolmas tunti

Ope-robotti

Palautetaan mieleen edellisellä tunnilla katsottu video voileivän valmistamisesta. Opettaja pyytää oppilailta näiden keksimiä toimintoja, joita voidaan suorittaa luokassa. On hyvin odotettavaa, että ideoita ei tule. Varaideana oppilaat saavat yhdessä laatia ohjeen, kuinka tussikirjoitus pyyhitään pois valkotaululta. Opettaja kirjoittaa oppilaiden luettelemat ohjeet vaiheittain tietokoneella siten, että oppilaat näkevät ne. Kun ohjeet ovat oppilaiden mielestä valmiit, ne testataan. Opettaja toimii tässä vaiheessa robottina. Se on aina oppilaista hyvin innostavaa. Ohjetta korjataan niin monta kertaa, että se on toimiva.

Esimerkki mahdollisesta algoritmista:

1. Nouse ylös tuolilta.
2. Kävele valkotaulun eteen.
3. Ota sieni käteesi.
4. Pyyhi sienellä kirjoitus taululta.

Algoritmien muodostaminen

Seuraavaksi kerrotaan oppilaille tunnin tavoitteet. Tavoitteena on:

- Oppia, mitä algoritmi tarkoittaa.
- Harjoitella yksiselitteisten käskyjen kirjoittamista.
- Oppia pilkkomaan toiminnot pieniin osiin.
- Harjoitella yhteistyötä muiden ryhmäläisten kanssa.

Tehtävän idea on poimittu Innokas-materiaalista https://www.innokas.fi/wp-content/uploads/2019/02/Ohjelmoinnillinen_ajattelu_-teht%C3%A4v%C3%A4kortti.pdf

Oppilaille annettava tehtävä 1

Keksikää erilaisia toimenpiteitä, joita koulussa voi tehdä. Valitkaa yksi toimenpide, jolle suunnitellaan algoritmi eli vaiheittaiset toimintaohjeet. Myös valmiita ideoita saa käyttää:

- Luokan ikkunan avaaminen, huutaminen ulos ”moi” ja ikkunan sulkeminen
- Kirjan hakeminen omasta laatikosta
- Nopan hakeminen noppalaatikosta ja nopan pyöräyttäminen pöydällä

Tehtävä 2

Kirjoittakaa valitsemanne toiminnan jokaisen vaiheen ohje omalle paperisuikaleelle. Kun olette valmiita, testatkaa ohjeet. Lisätkää ohjeita väleihin tarvittaessa. Testatkaa ohjeet uudestaan. Arvioikaa, ovatko ne toimivat. Numeroikaa vaiheet oikeassa järjestyksessä, kun ohjeet ovat mielestänne toimivat.

Tehtävä 3

Kokeillaan noudattaa muiden ryhmien algoritmeja. Toimintojen ohjeiden kokeilun jälkeen pohditaan:

- Olivatko vaiheet ja käskyt oikeassa järjestyksessä?
- Millaiset käskyt toimivat? Entä millaiset eivät?
- Millaisia muutoksia täytyy palata tekemään?
- Annetaan jokaiselle ryhmälle palautetta siitä, missä he onnistuivat erityisen hyvin.

Jatkotyöskentely

Seuraavassa vaiheessa siirrymme Scratchin ohjelmointiharjoituksiin.

Pistetyöskentelyä Scratchin ja BeeBottien parissa, 4.lk

Emmi Mykkänen, CC BY-SA 4.0

26.9.2020

TUNTISUUNNITELMA

LUOKKA-ASTE: 4.LK

AIHEPIIRIN VALINTA JA RAJAUS

Oppituntien aiheena on ohjelmointi eri menetelmien avulla. Oppitunnit ovat matematiikan tunneilla. Tarkoitus on harjoitella koodausta visuaalisen ohjelmointiympäristön parissa sekä toiminnallisesti. Toteutetaan tunnit pistetyöskentelynä. Tarkoituksena on tutustua Scratch:iin, beebotteihin, kaverin ohjelmointiin ja tangram- työskentelyä. Tavoitteena on tuoda ohjelmointi, koodaus ja algoritminen ajattelu lähelle oppilasta. Harjoitellaan samalla yhdessä työskentelyä niin ryhmän kuin parinkin kanssa. Lisäksi oppilas harjoittelee työskentelemään itsenäisesti.

ESITIEDOT OPPILAILLE JA OPETTAJALLE

Oppilas: Scratch- pisteellä työskentelyä varten oppilaalla olisi hyvä olla hallussa perustiedot ohjelmasta. Jos tuntia haluaa käyttää alkulämmittelynä ohjelmointiin, voisi opettaja luoda valmiit tiiviit ohjeet työskentelypisteelle. Muille pisteille oppilas ei tarvitse ennakkotietoja.

Opettaja: Scratch olisi hyvä olla opettajalle tuttu ympäristö, jotta opettaja voisi ohjeistaa pisteelle harjoituksen tai laatia ohjeet työskentelylle. Muutoin ei ennakko-osaamista tarvita.

VALITTUUN AIHEPIIRIIN LIITTYVIEN OPPIMISTAVOITTEIDEN MÄÄRITTELY

Tavoitteet perustuvat peruskoulun opetussuunnitelman perusteisiin ja oppilaan työskentelyyn. Osatavoitteista perustuu ohjelmoinnin oppimiseen ja osa työskentelyn harjoitteluun. Tehtävät tehdään matematiikan tunneilla, joten tavoitteita arvioidaan matematiikan alaisuudessa.

- Oppilas osaa selittää, mihin koodaamista voi käyttää.
- Oppilas harjoittelee koodaamisen alkeita.
- Oppilas toimii ryhmänsä aktiivisena jäsenenä ja kuuntelee muita.
- T14 innostaa oppilasta laatimaan toimintaohjeita tietokoneohjelmina graafisessa ohjelmointiympäristössä (POPS 2014)

ARVIOINTI

Opettaja:

- Arvioi havainnoimalla oppilaiden toimintaa ja osallistumista.
    - Osallistuuko oppilas työskentelyyn? Miten?
    - Toimiiko osana omaa ryhmäänsä?
    - Käy läpi myös oppilaan itsearvioinnin.
- Oppilas:
    - Oppilas tekee itsearvioinnin, jossa pohditaan omaa osallistumista työskentelyyn ja muiden huomioonottamista.

TYÖSKENTELYVÄLINEET JA OPETUSMENETELMÄT

VÄLINEET:

- Tarvitaan pienryhmän koon mukaan läppärit (3-5kpl)
- Ohjeet työpisteille
- Beebotteja (3-5kpl) ja niille sokkeloreitti (esim. teipistä, legoista, yms.)
- Tangram-palat (3-5kpl) ja ohjekuviot
- Kaverin ohjelmointiin muutama este temppurataan (esim. hulavanne, hyppynaru, pari penkkiä)

OHJEET:

Oppilaat jaetaan pienryhmiin, noin 3-5 oppilasta per ryhmä. Käydään yhdessä läpi työskentelypisteet ja ohjeet yhteisesti. Tämän jälkeen oppilaat saavat siirtyä heille ohjattuihin pisteisiin ja aloittaa työskentelyn. Työskennellään yhdellä pisteellä puoli oppituntia. Yhteensä siis työskennellään pistetyöskentelynä 2 oppituntia.

1. piste: Scratch
   - Kirjaudutaan Scratch-sivulle omilla tunnuksilla. Aloitetaan joko itsenäinen ohjelmoinnin suunnittelu ja koodaus tai valitaan yksi valmis peli, jota koodataan ohjeen mukaan. Lopuksi muistetaan tallentaa oma tuotos.
   - Opettaja voi etsiä valmiita peliohjeita esimerkiksi täältä: https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn
2. piste: Beebotit
   - Opettaja auttaa työskentelyn käyntiin. Näytetään, mikä on Beebotti ja mitä sillä voidaan tehdä (liiketoiminnot: eteen, taakse, oikealle ja vasemmalle)
   - Annetaan vapaata kokeilua
     - Voi vaikka suunnitella koko ryhmän kesken hienon liikeyhdistelmän beebottien välityksellä
   - Tai voidaan ohjelmoida Beebotti kulkemaan sokkelon läpi.
     - Opettaja on luonut yhden sokkelon valmiiksi lattiaan.
     - Lisäksi voidaan antaa oppilaille mahdollisuus luoda sokkelo myös itse.
3. piste: Tangram-palat
   - Jokaiselle on pöydällä omat tangram-palat ja yhteinen nippu erilaisia kuvioita, joita koota.
   - Oppilaat voivat tehdä kuvioita itsenäisesti tai yhdessä parin tai koko ryhmän kesken.
   - Jos halutaan lisätä pientä pelillisyyttä, kuka ryhmä onnistuu tekemään eniten kuvioita.
   - Tangram-palat voi halutessaan askarrella itse. Netistä löytää paljon kuvioita, joita tulostaa.
4. piste: Kaverin ohjelmointi
   - Opettaja on luonut oppilaille temppuradan käytävään (tai omaan luokkaan tilan mukaan). Radalla tulee alittaa hyppynarulla tehty puomi, kulkea hulavanteen läpi, pujotella penkit/kartiot jne.

- Jakaudutaan pareihin. Toinen sulkee silmät ja toinen ohjeistaa kaverin temppuradan läpi. Vaihdetaan pareja. Oppilaat joutuvat miettimään, mitä kaikkea kaverille pitää muistaa sanoa, jotta selvitään radan läpi. Lopuksi voidaan pohtia, mikä auttoi suoriutumisessa ja mikä loi haasteita. Kirjataan muutama idea paperille ylös ja annetaan opettajalle.

MOTIVAATIO JA ROHKAISU:

Tuntien työskentelypisteet ovat vaihtuvia, leikillisiä ja pelillisiä. Oppilaat pääsevät itse työskentelemään toiminnallisesti yhdessä. Oppilaita voi motivoida yhteistyöhön pieni kilpailullisuus esimerkiksi tangram-palojen yhteydessä tai temppuradan pääseminen mahdollisimman nopeasti. Yhteistyöllä päästään paljon suurempii lopputuloksiin. Jos jollakin oppilaalle on paljon omaa kokemusta, voi hän esimerkiksi auttaa Scratch-työskentelyssä muita.

Oppilaita voidaan rohkaista myös jatkamaan ohjelmoinnin harjoittelua kotona erilaisilla ohjelmointisivustoilla tai muilla algoritmista ajattelua kehittävillä tehtävillä. Esimerkiksi tangram-paloille on olemassa monia erilaisia tehtäviä. Voidaan myös korostaa kaikkia ohjelmoinnin etuja opiskelun kannalta ja mahdollisen tulevaisuuden työn kautta. Vaikka pieni yhteinen keskustelu toisen oppitunnin lopuksi, jossa keskustellaan ohjelmoinnin ja koodauksen hyödyistä yleisesti.

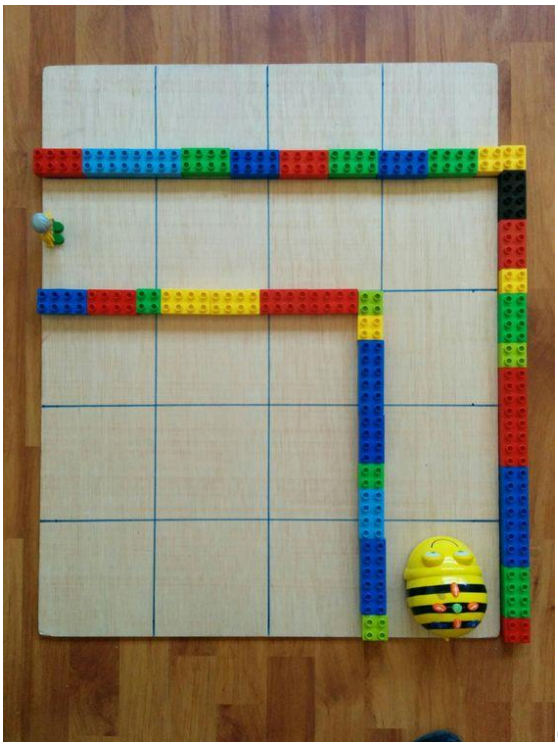
ITSEARVIOINTILOMAKE

Nimi: _____

Valitse hymiö kuvaamaan omaa suoriutumistasi.

				
1. Noudatin ohjeita.				
2. Jaksoin yrittää, vaikka en heti onnistunut.				
3. Osasin työskennellä ryhmässä.				
4. Otin kaikki mukaan työhön.				
5. Sain tehtyä oman pelin tai ohjelmoinnin.				
6. Kehuin kaveria.				
7. Sain tehtävät tehtyä.				
8. Tein työt huolellisesti.				

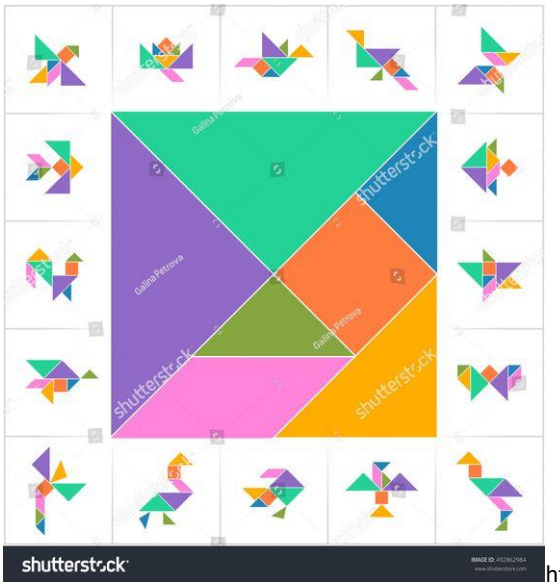
BEEBOTTI REITTI ESIMERKIKSI


(https://fi.pinterest.com/pin/9710955433768640/)

TANGRAM- MALLI JOS JOLLEKIN TÄYSIN VIERAS:

”…sisältää seitsemän palaa, joista voi yhdistellä lukemattomia eri kuvioita. Kappaleisiin kuuluu seitsemän geometrista muotoa: viisi kolmiota, joista kaksi suurta, yksi keskikokoinen ja kaksi pientä sekä yksi suunnikas ja yksi neliö. Kaikki kappaleet yhdessä muodostavat suuren neliön.” (Wikipedia)

Ajatuksena siis koota erilaisista paloista kuvioita.



https://fi.pinterest.com/pin/233413193176200099/

Lähteet:

https://fi.wikipedia.org/wiki/Tangram

https://www.oph.fi/sites/default/files/documents/perusopetuksen_opetussuunnitelman_perusteet_2014.pdf

https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn

Kuvat:

https://fi.pinterest.com/pin/233413193176200099/

https://fi.pinterest.com/pin/9710955433768640/

Ohjelmoinnin suunnitelma, 4.lk

Suvi Kaukovalta, CC BY-SA 4.0

6.1.2020

Ohjelmoinnin suunnitelma 4.lk, 15 oppilasta

Aihepiirin valinta ja rajaus

Suunnittelin ohjelmoinnin kokeiluni matematiikan tunneille, sillä ajattelun taitojen ja ongelmaratkaisutaitojen kehittäminen kuuluvat myös matematiikan opetussuunnitelmaan. Tunnit voidaan pitää peräkkäin samana päivänä tai yksitellen eri päivinä. Oppilailla ei ole juurikaan aikaisempaa kokemusta ohjelmoinnista eikä ohjelmointiympäristöistä. Oppilaat ovat tehneet alkuopetusvuosinaan muutamia harjoitteita, joissa ohjasivat opettajaa ja toisiaan sanallisesti. Oppilailla on tämän syksyn aikana ollut useampia ongelmanratkaisutunteja.

Näiden kahden oppitunnin (2 x 45min.) aikana oppilaat tutustuvat algoritmiseen ajatteluun ja visuaaliseen ohjelmointiympäristöön sekä oppivat ohjelmoinnin peruskäsitteet (lause/käsky ja muuttuja).

Ensimmäisen tunnin tehtävät suoritetaan ilman laitteita ja toisella tunnilla tutustutaan visuaaliseen ohjelmointiympäristöön Scratchin avulla.

Ensimmäinen tunti

Tunnin tavoitteet

- Oppilas ymmärtää, mitä algoritmi tarkoittaa.
- Oppilas harjoittelee kirjoittamaan ja muokkaamaan algoritmia.
- Oppilas kehittää algoritmista ajatteluaan ja yhteistyötaitojaan toimiessaan yhteistyössä muiden oppilaiden kanssa.

Virittäytyminen

Katsotaan yhdessä video (https://www.youtube.com/watch?v=j-6N3bLgYyQ), jossa isä noudattaa kirjaimellisesti lastensa ohjeita voileivän valmistamisesta. Videon katsomisen jälkeen annetaan oppilaiden muutama minuutti keskustella videon herättämistä ajatuksista. Tämän jälkeen keskustellaan yhdessä videon herättämistä ajatuksista ja kysytään oppilailta, mitä isä halusi videolla opettaa lapsilleen. Missä muualla tällaisia tarkkoja ohjeita tarvitaan? Johdetaan oppilaiden vastauksia apukysymyksillä ohjelmoinnin suuntaan.

Seuraavaksi kerrotaan oppilaille tunnin tavoitteet.

- Oppia, mitä algoritmi tarkoittaa.
- Harjoitella yksiselitteisten käskyjen kirjoittamista.
- Harjoitella yhteistyötä muiden ryhmäläisten kanssa.

Ope-robotti

Opettaja esittää robottia, joka tottelee oppilaiden suullisia käskyjä yksi kerrallaan. Oppilaat antavat robotille käskyjä vuorotellen siten, että robotti onnistuu pukemaan päälle ulkovaatteet. Lopuksi pohditaan, kuinka oppilaat onnistuivat ohjelmoinnissa. Toteuttiko, robotti toimenpiteet ilman virheliikkeitä? Millaisilla käskyillä päästiin haluttuun lopputulokseen ja millaiset käskyt eivät toimineet?

Algoritmin muodostaminen

Suunnitellaan yhdessä algoritmi eli vaiheittainen työohje kirjojen pakkaamiseen pöydältä reppuun ja testataan käytännössä sen toimintaa.

1. Oppilaat miettivät pienryhmissä aluksi, mistä suuremmista vaiheista repun pakkaaminen koostuu. Otsikoikaa ne paperille oikeassa järjestyksessä. Muistakaa jättää riittävästi tyhjää tilaa jokaisen vaiheen alle.
2. Numeroikaa jokaisen vaiheen alle omin sanoin oikeassa järjestyksessä yksiselitteiset käskyt, kuinka kyseinen vaihe voitaisiin toteuttaa.
3. Kokeilkaa noudattaa algoritmia itse.

Esimerkki mahdollisesta algoritmista:

- Repun avaus
    1. Nouse seisomaan.
    2. Siirry reppuusi viereen.
    3. Ota vasemmalla kädellä kiinni repun yläreunasta vetoketjun vierestä.
    4. Ota oikealla kädellä kiinni repun ison taskun vetoketjun vetimestä ja avaa vetoketju kokonaan auki.
- Kirjojen nostaminen
    1. Irrota molemmat kädet repusta.
    2. Käänny pöytääsi päin.
    3. Ota oikealla kädellä kiinni kirjojen alareunasta.
    4. Nosta kirjat ilmaan.
- Kirjojen reppuun laittaminen
    1. Käänny reppuusi päin.
    2. Ota vasemmalla kädellä kiinni repun etuosan yläreunasta.
    3. Laske oikeassa kädessä olevat kirjat reppuun yläreuna edellä.
    4. Irrota otteesi kirjoista.
- Repun sulkeminen
    1. Ota vasemmalla kädellä kiinni repun ison taskun vetoketjun siitä päästä, jossa vedin sijaitsee.
    2. Ota oikealla kädellä kiinni ison taskun vetoketjun vetimestä.
    3. Vedä oikealla kädellä ison taskun vetoketju kiinni.
    4. Jos vetoketjun vedin pysähtyy, siirrä vasen käsi repun yläreunaan vetoketjun viereen.
    5. Kun vetoketju on kokonaan kiinni, irrota otteesi repusta.

Lopuksi verrataan ja pohditaan koko luokka yhdessä ryhmien algoritmejä. Olivatko vaiheet ja käskyt oikeassa järjestyksessä? Millaiset käskyt toimivat ja millaiset eivät? Millaisia muutoksia täytyi tehdä, jotta päästiin haluttuun lopputulokseen?

Tunnin koonti

Keskustellaan oppilaiden kanssa, mitä he ovat oppineet algoritmin tekemisestä. Mikä algoritmi on? Mitä asioita algoritmia kirjoittaessa täytyy huomioida? Kerrataan, mikä oppilaista tuntui haastavalta ja miten haasteita saatiin ratkaistua yhdessä muiden kanssa.

Lopuksi toiminnallisella itsearvioinnilla (erinomaisesti – oppilas nousee seisomaan kädet ylhäällä, hyvin – oppilas seisoo kädet alhaalla, jonkin verran – oppilas istuu, tarvitsen vielä harjoitusta – oppilas käy kyykkyyn) oppilaat arvioivat oppistaan:

- Osaan omin sanoin kertoa, mikä algoritmi on.

- Osasin kirjoittaa täsmällisiä ohjeita algoritmitehtävässä.
- Ymmärrän, miksi käskyjen tulee olla yksiselitteisiä algoritmeissa.

Lähteet:
Tunnin tehtävät on muokattu Innokas! -materiaalin tehtävistä https://www.innokas.fi/wp-content/uploads/2019/02/Ohjelmoinnillinen_ajattelu_-teht%C3%A4v%C3%A4kortti.pdf

Toinen tunti

Tunnilla tarvitaan tietokone tai tablet-laite joka toiselle oppilaalle. Työskennellään pareittain saman laitteen äärellä tukien näin oppilaiden ajattelun taitojen kehittymistä.

Tunnin tavoitteet

- Oppilas harjoittelee ohjelmointia visuaalisessa ohjelmointiympäristössä.
- Oppilas harjoittelee ongelmanratkaisua yhdessä parin kanssa ja kehittää yhteistyötaitojaan.

Virittäytyminen

Oppilaiden tullessa luokkaan taululle on heijastettuna alla oleva kuva Scratchin ohjelmointiympäristöstä.

Aloitus

Kerrotaan oppilaille tunnin tavoitteet:

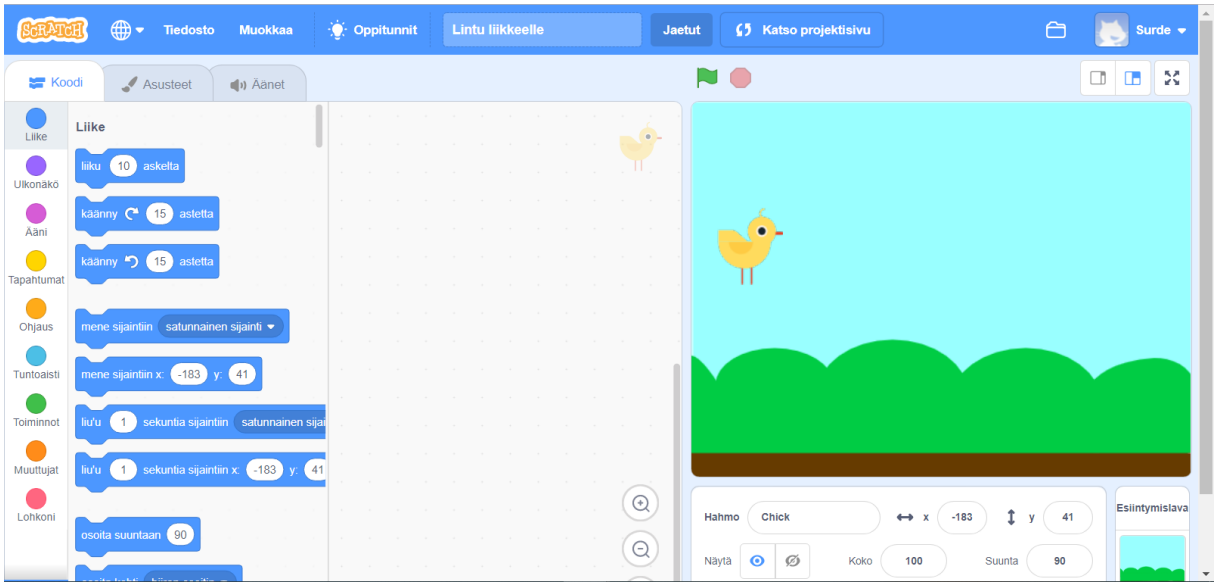
- Harjoitella ohjelmointia visuaalisessa ohjelmointiympäristössä yhdessä parin kanssa.
- Oppia, mitä tarkoittavat ohjelmoinnissa käytetyt termit lause/komento ja muuttuja.
- Harjoitella kirjoittamaan käskyjä muuttujalle.

Kertaus

Kerrataan oppilailta kysellen, mitä algoritmi tarkoittaa ja mitä käskyjen kirjoittamisessa täytyy huomioida. Muistellaan edellistunnin tehtäviä.

Ohjelmointiympäristön esittely

Näytetään oppilaille Scratch-sivusto ja miltä ohjelmointiympäristö näyttää.



Opetetaan, mistä löytyy komennot, joista lause muodostetaan ja mistä saadaan muuttuja eli tässä tapauksessa lintuhahmo, jota liikutellaan. Näytetään, kuinka komentoja saa valittua ja raahattua koodialueelle ja kuinka näyttämöalueella nähdään, kuinka koodi toimii.

Ohjelmointi

Ohjeet oppilaille

1. Mene osoitteeseen https://scratch.mit.edu/projects/348012089/ ja klikkaa oikealta sinistä Katso sisälle -painiketta.
2. Sivun yläreunasta maapallosta vaihda kieleksi Suomi.
3. Etene alkuun opettajan ohjeiden mukaan.
4. Tämän jälkeen kokeile rohkeasti eri toimintoja/käskyjä lauseeseen ja katso, miten muuttujan toiminta muuttuu. Voit halutessasi kokeilla lisätä myös toisen hahmon ympäristöön.

Kun kaikki oppilaat ovat päässeet ohjelmointiympäristöön ja saaneet kielen vaihdettua suomeksi, opettaja voi kysellä, mitä meidän tulikaan tehdä.

- Voidaan aloittaa liikkeestä, koska hahmo tuli saada liikkeelle.
    - Lisätään liiku 10 askelta –palkki koodialueelle. Kokeillaan, toimiiko koodi. -> ei toimi.
- Opettaja ohjeistaa, kuinka ohjelmointilause saadaan alkuun.

    - Lisätään tapahtuma, josta toiminta/liike alkaa. Esim. kun painetaan välilyönti –palkki.
- Ohjeistetaan, kuinka hahmo saadaan liikkumaan takaisin päin.
    - Lisätään liiku 10 askelta –palkki. Muutetaan arvoksi -10 (#takaisin päin akselilla) Kokeillaan toimiiko koodi -> ei toimi. Miksi? (#toiminnot tapahtuvat yhtä aikaa)
    - Lisätään odota –palkki.
- Ohjeistetaan oppilaita rohkeasti kokeilemaan erilaisia käskyjä ja mahdollisesti lisäämään myös toinen hahmo alustalle.
- Kannustetaan ratkomaan parin kanssa vastaan tulevia haasteita ja ongelmia.

Opettaja havainnoi työskentelyä ja toimii ohjaajana taustalla. Oppilaiden harrastuneisuus pääsee näkyviin heidän oman työskentelynsä aikana. Samalla pareittain työskennellessä oppilaat yhdessä ratkovat vastaantulevia ongelmia ja myös opettavat toisiaan.

Koonti

Esitellään taululle heijastaen oppilaiden aikaansaannoksia ja keskustellaan niistä. Lisäksi keskustellaan, mikä oli haasteellista ja mikä helppoa ja kuinka oppilaat löysivät ratkaisuja kohdattuihin ongelmiin. Lopuksi kerrataan vielä, mitä tarkoittavat algoritmi, lause ja muuttuja sekä millaisia käskyjä tietokoneet lukevat/”ymmärtävät”.

Arviointi

Opettajan havainnointi ja havaintojen kirjaaminen ylös tunnin aikana.

Tunnin lopussa keskustelu, miltä työskentely tuntui ja mitä opittiin.

Lähteet

Seuraavista lähteistä otettu ideoita ja vinkkejä toisen tunnin suunnitteluun

- Pulmaario-ohjaajan opas
- https://mooc.helsinki.fi/pluginfile.php/80174/mod_resource/content/9/ohjelmointiin-ja-internetiin-liittyvia-harjoituksia-projekteja-ja-tavoitteita-eri-luokka-asteille.pdf

Ohjelmointikokeiluja Scratchilla, 4.lk

22.11.2020

Aihe

Kahden tunnin aikana on tarkoitus tutustuttaa 4. luokan oppilaat opetussuunnitelman perusteiden mukaisesti graafiseen koodaukseen Scratchin avulla. Tunnit pidetään matematiikan tunneilla ja koronasta johtuen etänä.

Aiemmat tiedot

Oppilaat ovat aiemmin tutustuneet koodauksen matematiikan kirjan tehtävien, erilaisten toiminnallisten tehtävien sekä kynällä ja paperilla tehtävien koodausharjoitusten avulla.

Opettaja on aiemmin käyttänyt Scratchiä.

Oppimistavoitteet

Laaja-alaisena tavoitteena on opettaa ajatteluntaitoja sekä tieto- ja viestintäteknologian käyttöä.

Tutustua koodauksen alkeisiin graafisessa järjestelmässä.

Harjoitella koodauksen alkeita graafisella järjestelmällä.

Arviointi

Oppilas ottaa kuvan tuotoksestaan ja palauttaa sen classroomiin opettajalle, joka kommentoi sitä. Tehtävän palautuksen yhteydessä oppilas ohjataan kuvailemaan tuntemuksiaan ohjelmointitehtävästä kommenttikenttään. Opettaja arvioi oppilaan työskentelyä kommenttikeskusteluiden ja lopputuotoksen avulla. Lopuksi vielä käydään yhteinen loppukeskustelu koko luokan kanssa Meet- ympäristössä.

Tarvikkeet

Oma kone

Internet yhteys

Tunnin kulku

1. tunti

Opettaja on tehnyt tunnin valmiiksi classroom ympäristöön, joka on oppilaille tuttu. Sinne opettaja on luonut tehtävän, jonka yhteyteen hän on **kuvannut valmiiksi opetusvideot**. Tehtävän yhteydessä on myös linkki Scratch -sivuille.

Oppilaille annetaan classroomissa ohje:

Katso **Scratchiin kirjautuminen** video

- Kirjaudu Scratchiin ohjeen mukaan, pyydä tarvittaessa aikuiselta apua

- Ohjelman käyttö vaatii rekisteröitymistä, mutta ei kysy esimerkiksi osoitetta.

- Voit käyttää rekisteröitymiseen koulun sähköpostitunnustasi.

- Ohjelmaa ei tarvitse ladata koneelle, vaan sitä voi käyttää internetissä.

Scratchin peruselementit videolla on selitetty ohjelman alkunäytön eri osien toiminta.

Koodausta Scratchilla - harjoitus 1 -videossa näytetään, miten Scratchilla voi koodata.

- Tee oma ensimmäinen koodausharjoituksesi. Sinun ei tarvitse tehdä samoja kuin videossa, vaan voit luoda vapaasti oman.

- Ota kuva omasta tuotoksestasi ja lataa se tehtävän yhteyteen!

Tunnin aikana opettaja päivystää sovitulla Meet -kanavalla, josta oppilaat voivat käydä kysymässä neuvoa.

Tässä vaiheessa oppilailta ei vielä odoteta paljoakaan. Heidän täytyy lähinnä saada aikaan jonkinlainen visuaalinen tuotos, jossa on tausta ja hahmo. He voivat toistaa perässä opettajan videolla näyttämät jutut tai jos taidot riittävät voi tehdä täysin omanlaisen kuvan.

2. tunti

Opettaja on luonut Classroomiin uuden tehtävän. Tehtävän yhteydessä on taas linkki Scratch -sivuille.

Oppilaille annetaan ohje:

Katso Scratch –sivustolta jokin opetusvideo ja tee sen oppien pohjalta toinen koodauskokeilusi. Ota tuotoksestasi kuva ja liitä se tehtävän yhteyteen.

Tässä on ideana, että oppilaat valitsevat oman mielenkiintonsa pohjalta Scratchin omista opetusvideoista mieluisan, jonka oppien pohjalta kokeilevat koodata oman tällä kertaa toiminnallisuutta sisältävän videon. Palautukseksi riittää kuitenkin kuva lopputuotoksesta eli opettaja ei pääse vielä arvioimaan toiminnallisuuden toteutusta.

Pohdintaa:

Tehtävä toteutetaan olosuhteiden pakosta etänä, mutta uskon sen toimivan kohtuullisen hyvin. Jokaisella on käytössä laitteet ja ainakin jokaisella on nyt mahdollisuus toteuttaa tehtävä itsenäisesti ilman häiriötekijöitä luokassa. Etäolosuhteista johtuen yhteistyöhön ei juurikaan ole mahdollisuuksia. Toki ei ole kiellettyä, että oppilaat keskustetelevat tehtävästä omissa ryhmissään Meetissä. Ehkä ainakin osalla herää innostus tutustua ohjelmointiin myös koulutehtävien ulkopuolella. Osa oppilaista odottaakin tehtävää innolla ja heillä on jo aiempaa kokemusta Scratchin käytöstä.

Scratch-ohjelmointiin tutustumista, 4.lk

25.10.2020

Koodauksen alkeita 4 lk

Kyseessä vilkas15 oppilaan 4. lk, joilla on TVT-tunti kerran viikossa. Kyseessä ns. vieras luokka, jolle opetan vain tämän TVT:n ja yhden musiikkitunnin viikossa. Oppilailla ei ole kokemusta Scratch-ohjelmoinnista, tämän olen selvittänyt etukäteen. Arvioin oppilaiden työskentelyintoa, kokeiluhalukkuutta sekä valmista 'peliä', jotka oppilaat saavat esitellä toisilleen. Tavoitteena on saada innostus koodaukseen ja vähimmäistavoite on saada oma pelihahmo liikkeelle nuolinäppäimistä. Kaikki, mitä tulee lisää on bonusta 😊. Käytössä ovat chromebookit.

Ensimmäinen tunti:

- Palauttelemme mieliin viime kerran hässäkän, joka nousi siitä, etteivät oppilaat päässeet kirjautumaan Wilmaan. Kyseessä oli heidän ensimmäinen kertansa, jolloin heidän piti kirjautua annetulla henkilökohtaisella numero-kirjain-merkkiyhdistelmällä. Iso hämmästys nousi, kun open naputtelemana salasana kelpasi. Mistä mahtoi johtua? Miksei kirjautuminen onnistunut, jos sarjaan tuli yksikin väärä merkki? Tavoitteena saada oppilaat huomaamaan, ettei kone tee muuta kuin mihin se on ohjelmoitu. Se ei ymmärrä ison kirjaimen tarkoittavan ehkä pientä jne. (n.5 min)
- Seuraavaksi katsomme videon, jossa isä tekee leivän tismalleen lasten antamien ohjeiden mukaan https://www.youtube.com/watch?v=j-6N3bLgYyQ (n. 5 min)
- Keskustelu oppilaiden tekemistä huomioista
- Seuraavaksi oppilaat antavat minulle ohjeet siitä, kuinka otan repustani omenan, käyn pesemässä sekä käteni että omenan ja syön sen. Pyrin toimimaan tismalleen ohjeiden mukaan. Tunnilla on mukana ohjaaja, joka voi jakaa oppilaille puheenvuorot minun toimintani siitä häiriintymättä. (keskustelu ja opettajan koodaus n. 10 min) (Olipa aika lystiä!)
- Lopputunnista tutustumme schratchiin. Katsomme yhdessä, mistä saa valittua hahmot ja mitä erilaisia käskyjä ohjelmassa on valmiina. Kerron, että alkuun laitetaan palikka, jossa on vihreä lippu ja että oppilaat voivat itse kokeilla, kuinka saisivat hahmon liikkeelle. Toisia saa neuvoa, toisilta ja minulta saa kysyä. Loppupuolella oppilaat saavat kertoa, mitä toimintoja saivat aikaan. (n. 15-20 min) (Oppilaat saivat aikaan ääniä, puhekuplia ja pyörimistä. Myös pyrähtelyjä random-paikkoihin)
- Ennen tunnin loppua kehotan oppilaita keskustelemaan kotona kirjautumisesta scratchiin, koska se tarvitsee vanhemman sähköpostiosoitteen.
- Lopuksi desinfioimme ja palautamme koneet latauskaappiin. (n. 5 min.)

Toinen tunti:

- Aloitamme uuden scratch- session edelliskertaa ohjatummin. Tavoitteena saada hahmo liikkumaan nuolinäppäimistä. (Niin luulin, pakko lisätä mitä todellisuudessa tapahtui: myrskyn aiheuttama sähkökatko, älytaulu ei siis toiminut, koneisiin jaettiin nettiä sekä open että oppilaiden puhelimilla.. Toisaalta hyvää harjoitusta toimia pelkästään kuullun ohjeen avulla, lujaa keskittymistä vaati sekä opettajalta että oppilailta)
- Muistelemme, mitä toimintoja viime viikolla saimme aikaan. Kukaan ei keksinyt, kuinka hahmon liikkumista voi ohjata, jottei se liiku koko aikaa täysin saman kaavan mukaan. Kerron, että tarvitsemme tietoa x – ja y-akselista, palautan koordinaatiston oppilaiden mieliin. (Tähänkin lisäys: koordinaatisto oli heille täysin vieras asia, ei ollut tullut heidän matematiikan tunneillaan viime vuonna esille lainkaan. No, opettelimme tätä hiukan.)
- Kerroin, että nyt laitamme hahmon liikkumaan nuolinäppäimillä. Palautamme mieliin, että aloitamme lipusta. Pyydän oppilaita katsomaan ohjaus- ja liikepalikoita, mitä tarvitsemme, jotta

liikkuminen tapahtuu nuolista. Neuvon ehdollisen komennon, ’jos välilyönti, niin..’ ja kuinka siihen vaihdetaan nuoli oikealle. Entäpä mitä tarvitaan, jotta kaveri liikkuu oikealle siitä? Pyrin saamaan pohdintaa ja keskustelua aikaan, annan kokeilla ja yrittää. Lopulta katsomme ensimmäisen palikan yhdessä niin, että hahmo liikkuu oikeasta nuolinäppäimestä jonkin verran oikealle. Samaan tapaan keskustellen, yrittäen ja oivaltaen oppilaat saavat koodata hahmon liikkumisen vasemmalle, ylös ja alas.

- Mikäli aikaa riittää, lisäämme hahmon, jota ensimmäinen hahmo ajaa takaa. Opettelemme koodaamaan osumasta pisteet ja näin saamme jonkinlaisen tuntuman yksinkertaisen pelin tekemiseen.
- Lopuksi tallennamme projektin koneille ja jatkamme seuraavalla kerralla.
- Pelien valmistuttua testaamme niitä yhdessä. Oppilaat saavat kertoa omista peleistään ja siitä, miten saivat pelinsä valmiiksi. Selvittivätkö kokeilun ja erehdyksen kautta, saivatko apua kavereilta/opelta jne. He saavat vielä kertoa, miten tyytyväisiä ovat omaan tuotokseensa ja haluaisivatko tehdä scratchilla jatkossa jotain muuta.

Algoritminen ajattelu kasvin tunnistuksessa, 5.lk

Timo Hartus, CC BY-SA 4.0

12.4.2021

Tuntisuunnitelma: Algoritminen ajattelu kasvin tunnistuksessa

Aihepiirin valinta ja rajaus

Missä oppiaineessa ja mihin aihepiiriin haluat soveltaa ohjelmointia ja miksi?

Biologia, peruskoulun 5. luokka, kasvien tunnistus. Kasvien tunnistuksessa sovelletaan algoritmista ajattelua ja systematiikkaa. Systemaattinen ajattelu algoritmien avulla helpottaa eri kasvilajien tunnistamista ja ryhmittelyä.

Minkälaisia ilmiöitä valitsemaasi aihepiiriin liittyy?

Kasvien kehitys ja kasvaminen kasvukauden aikana sekä näkyviin tulevat erot kasvien ulkonäössä.

Minkälaisia yksittäisiä oppimistavoitteita näihin ilmiöihin tai niiden havainnointiin liittyy?

Kasvien tunnistamisen ja luokittelun oppiminen ja kehittyminen sekä systemaattisen algoritmisen ajattelun kehittyminen kasvitunnistuksessa ja yleisellä tasolla myös.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Opitaan tunnistamaan tietyn systematiikan avulla kasveja niiden rakenteiden ulkoisten ominaisuuksien perusteella.

Millaisia oppimistavoitteita asetat oppikokonaisuudelle toisaalta ohjelmointiin ja toisaalta kohdeoppiaineeseen liittyen?

Tavoitteena on oppia algoritmisen ajattelun soveltamista eri aiheisiin, tässä tapauksessa kasvien tunnistamiseen. Pyritään oppimaan systematiikkaa, joka helpottaa asioiden oppimista ja muistamista. Tämä harjoitus kehittää siis myös kasvien tunnistusta ainakin jonkin verran.

Miten opettaja tai joku muu arvioi opiskelijan osaamista? Arvioidaanko sekä ohjelmointiosaamista että kohdeoppiaineen osaamista? Kokonaisuutena vai toisistaan erillään?

Tehtävässä opettaja tarkkailee oppilaiden motivoitumista tekemiseen ja positiiviseen kokemukseen, joka edistää myös jatkossa tapahtuvaa oppimista. Oppilaat voivat myös itse arvioida, osaavatko vai tarvitaanko lisäpetusta.

Miten oppilas arvioi omaa osaamistaan?

Oppilas voi itse testata pareittain tai pienryhmissä tehtyjen algoritmien avulla pystyykö käytettävissä olevien näytteiden tunnistamisen toteuttamaan muodostetuin

algoritmein. Oppilas saa onnistumisen kokemuksia, kun tunnistaminen lisää kasvien tuntemusta. tehtävää voi harjoitella opettajan ja oman pienryhmän avulla useita kertoja, tarpeen mukaan.

Mitä välineitä oppikokonaisuudessa käytetään?

Tietokoneet ja/tai muistiinpanovälineet, sekä kasvinäytteitä. Kasvinäytteet oivat olla paperikuvia, tietokoneella olevia digitoituja kuvia kasveista tai aiemmin kerättyjä kasvinäytteitä.

Minkälainen johdantomateriaali tai ohjeistus tehtäviin annetaan?

Opettaja esittää lyhyen, 15 minuuttia, alustuksen tunnin aiheena olevien kasvien rakenteista ja esimerkkejä niistä tavallisissa kasveissa.

Esimerkkeinä voidaan käyttää kukkakasvien sekä puiden ja pensaiden tunnistusta, joista tarkasteltaisiin pohdiskelussa mm. seuraavia ominaisuuksia yrittäen keksiä niiden tarkastelujärjestykselle algoritmit, jolla edetään ratkaisua.

Kukkakasvit:

- Kasvin löytöpaikkaa ja kukinta-aikaa koskevat ominaisuudet.

- Kukkaa ja kukintoa koskevat ominaisuudet.

- Lehteä tai lehdyköitä koskevat ominaisuudet.

- Muut tunnistamista helpottavat ominaisuudet, kuten korkeus, karvaisuus tai lehtivihreättömyys.

Puiden ja pensaiden tunnistaminen:

- lajin havaintopaikkaa ja kukinta-aikaa koskevat ominaisuudet

- Kukkaa ja kukintoa koskevat ominaisuudet

- Lehteä tai lehdyköitä koskevat ominaisuudet

- Oksaa ja silmuja koskevat ominaisuudet

- Hedelmää koskevat ominaisuudet

- Muut tunnistamista helpottavat ominaisuudet, kuten korkeus, piikikkyys ja kasvumuoto

Millä perusteilla oppilaat motivoituvat tästä aiheesta?

Kaikki tuntevat varmasti jonkin kasvin ja siitä on hyvä lähteä innostavasti selittämään kasvin osia ja algoritmien mahdollisuuksia systemaattiseen tunnistusprosessiin.

Miten opetuksessa rohkaistaan yhteistyötä?

Oppilaat toimivat pienryhmissä tai pareittain, jolloin harjoittelu sujuu innostavana vapaamuotoisena tapahtumana. Kaikkien pitää päästä tasapuolisesti osallistumaan. Opettaja valvoo tehtävää.

Miten oppikokonaisuuden aikana voidaan auttaa oppilaita tuomaan mahdollinen aiempi ohjelmointiosaamisensa (kenties toisista ympäristöistä ja työkaluista) oppikokonaisuuden kontekstiin?

Mahdollisena jatkotyönä tai lisäharjoituksena tietotekniikassa edistyneet ja innokkaat oppilaat voivat koota algoritmeja ja tunnistettavien näytteidenkuvia ohjelmaksi.

Miten oppikokonaisuuden aikana tai sen jälkeen voitaisiin tukea opitun asian yhdistämistä muihin tilanteisiin, joissa oppilaat kohtaavat ohjelmointia tai algoritmista ajattelua?

Opettaja selventää algoritmisen ajattelun systematiikan hyödynnettävyyttä eri aloilla, jolloin oppilaat voivat miettiä muita mahdollisuuksia eri aihealueilla algoritmisen ajattelun käyttämiseen.

Oppitunnin pituus ja aikataulutus

2 kertaa 45 minuuttia

1. Opettajan alus 10-15 minuuttia

2. Ryhmiin jako ja välineiden jako 5 min.

3. Kasvitunnistukseen sopivien algoritmien pohtiminen 20 min.

Tauko

4. Ryhmien ehdotukset ja keskustelua opettajan johdolla 10-15 min.

5. Edellä mahdollisesti käyttökelpoisten tunnistusalgoritmien testaaminen ja kasvin tunnistusharjoitukset kasvinäytteiden avulla. Toimitaan edelleen pienryhmissä tai pareittain, 20-25 min

6. Yhteenveto opettajan johdolla ja ajatukset tehtävän jatkosta, noin 5 min.

Jatkossa, esimerkiksi ohjelmoinnista innostuneet voisivat tehdä tunnistuspeliä, jossa toimittaisiin kehiteltyjen algoritmien mukaan edeten. Osa voi mahdollisesti kuvata kasveja materiaaliksi tms.

Ohesmateriaalia:

https://www.luontoportti.com/suomi/fi/kasvit/

BeeBot-kierrätyspeli, 5.lk

17.3.2021

Beebot-kierrätyspeli

Tavoitteet: kierrätys, toiminnallinen oppiminen, ryhmätyöskentely ja beebotin käyttö

Peli suunnitellaan pienryhmissä. Ensimmäisellä ympäristöopin tunnilla 5lk. Oppilaat piirtävät paperille kolme erilaista ”jätesäkkiä”. Kaikilla ryhmillä on eri ”jätesäkkejä”. Yhdellä on bio-, seka- ja pahvijäte. Toisella lasi- ja muovijäte jne. Oppilaat piirtävät ruudutetulle beebot-alustalle määrättyjä jätesäkkejä. Tämän jälkeen he tekevät jätekortteja.

Seuraavalla tunnilla päästään pelaamaan. Pelin ideana on nostaa kortti ja ohjata beebot oikealle jätesäkille, liittyen siihen mihin jätteeseen nostettu kortti kuuluu. Pisteen saa siitä että ohjaa beebotin oikealle jätesäkille.

Kun peliä on pelattu jää siitä valmis peli myös muille luokka-asteille. Pelin aikana voidaan samalla arvioida ryhmätyöskentelyn sujumista. Beebot robotit toimintaperiaatteena ovat jo ennestään tutut oppilaille, joten niiden käytön kertaamiseen ei tarvitse käyttää aikaa.

Scratch-projekteja pienryhmissä, 5. lk

16.4.2021

Scratchiin liittyvästä opetusvideosta (Sumu, 2019) mieleeni jäi scratchin yhteisölliseen kehittämiseen tähtäävä remiksaa-toiminto. Sen sijaan siis, että kaikki oppilaat tekisivät samanlaisen pelin ja loisivat esim. saman pallon ja maalin ja samat toiminnallisuudet, valittaisiin esim. pienryhmittäin eri pelit jostain valmiiden tekeleiden pankista ja testattaisiin, suunniteltaisiin muutos ja koodattaisiin se joko pari- tai ryhmätyönä.

Linkki visuaalinen ohjelmointityökalu Scratchiin: https://scratch.mit.edu/

Taustaoletukset:

- Oppilaat ovat jo edellisellä luokka-asteella kokeilleet ja tehneet jotain scratchilla tai ollaan tehty pienet peliohjelmat scratchilla ennen tätä oppituntia, jotta scratchin käyttö on jotenkuten tuttua ja testaaminen ja koodin muuttaminen raahaamalla jne olisi jo tuttua.
- Oppilailla on scratch-tunnukset, jotta pelien kommentointi onnistuu.
- Opettajan osalta: Opettajalla on vahva ryhmäntuntemus ja oppilaantuntemus, jotta heterogeenisten ja toimintakykyisten pienryhmien tai parien muodostaminen onnistuu mahdollisimman hyvin.
- Tarvittaessa koulunkäynninohjaajan tuki.

Oppitunnin (kaksoistunti) tavoitteena: - matematiikka ohjelmointi sekä työskentelytaidot, ryhmätyöt

- harjoitella visuaalista ohjelmointia:
- testata jo tehtyä yksinkertaista peliä
- sanallistaa pelin tapahtumat eli toiminnot “selkokielellä”
- sanallistaa pelin valmis koodi eli palikoiden tekemiset
- harjoitella ryhmätyönä pelin muutoksen suunnittelua
- ja valita yksi muutos toteutettavaksi
- toteuttaa muutos
- testata muutettua peliä ja selittää muulle ryhmälle/opettajalle tehty muutos
- Ylöspäin eriyttäminen: Nopeat ryhmät:

1. Neljän ryhmästä kaksi tai parityöskentelyssä toinen tekee valmiiseen ohjelmaan virheen ja toinen yrittää selvittää, mikä virhe oli ja miten virhe korjataan.
2. Jos koodissa on valmiita arvoja tai esim. koordinaatteja, testataan mitä tapahtuu, kun niitä arvoja muuttaa toiseksi

Arviointi:

- Voidaan hyödyntää itsearviointiportaita - kriteerit avoimesti ennen tuokiota oppilaille
- Arviointikriteeri 1: muutokset ohjelmaan

1. porras: Teimme yhden muutoksen valitsemaamme ohjelmaan

2. porras: Teimme kaksi tai useamman muutoksen ohjelmaan
3. porras: Teimme useampien muutoksien lisäksi virheenmetsästystä

- Arviointikriteeri 2: yhteistyö ryhmässä
    1. porras: ehdotukset ja päätökset teki pääasiassa toinen ryhmän jäsen
    2. porras: tein pääasiassa itse ehdotukset ja/tai päätökset ryhmässä
    3. porras: teimme vuorotellen ehdotuksia ja päätöksen yhdessä

- Arviointikriteeri 3: Pelin toimintojen selittämisen sanallistaminen
- Arviointikriteeri 4: Koodin selittämisen selkeys

Opetustuokion ajan kerätään esiintyviä käsitteitä muistioon/taululle/vihkoihin:

- muuttuja, liike, toiminto, silmukka, virheenmetsästys, debuggaus

Lähteet:

Sumu, Virpi. 2019.

https://youtu.be/qwWE2OiQj1s

Ohjelmointijakson aloitustunnit, 5. lk

Mari Lehikoinen, CC BY-SA 4.0

24.10.2021

Ohjelmointijakson aloitustunnit 5.-luokkalaisille

Suunnitelman laatija: Mari Lehikoinen

Tämä tuntisuunnitelma on tarkoitettu osaksi matematiikan opiskelua. Kohderyhmänä ovat 5.-luokkalaiset, jotka ovat satunnaisesti tutustuneet ohjelmointiin muun muass Viruskaappari-pelin, matematiikan oppikirjan koodaustehtävien sekä sähköisten Ville-tehtävien kautta. Oppitunnit toimivat aloituksena pidemmälle jaksolle, jossa harjoitellaan Scratch-ohjelmointia.

Tavoitteena näillä pohjatunneilla on

- ymmärtää, että tietokoneet toimivat ihmisten antamien käskyjen varassa
- ymmärtää, että algoritmi tarkoittaa käskyjonoa
- oppia purkamaan tapahtumasarja osiin
- muodostaa suullisesti algoritmi arjen tilanteeseen ja ohjelmoida sillä toista oppilasta
- tutustua Scratch-ohjelmointiympäristöön
- animoida oman nimen kirjaimet Scratchissa ja ymmärtää, että jokainen hahmo tarvitsee omat käskyt toimiakseen
- työskennellä parin kanssa toiminnallisesti, aktiivisesti osallistuen ja tasa-arvoisesti
- formatiivisen arvioinnin keinoin tukea ja rohkaista oppilaita ajattelussa, kokeilussa ja parityöskentelyssä

Oppitunneilla tarvittavat välineet:
1. tunnilla oppilaan oma matematiikan vihko ja kynä.
2. tunnilla tietokone jokaiselle oppilaalle. Opettajalle tietokone + videotykki.

TUNTI 1: Ohjelmoinnin perusteet toiminnallisesti

Paripohdinta tunnin aiheeseen johdattamiseksi (5min)
Keskustelkaa parin kanssa muutama minuutti aiheesta ”Onko tietokone viisas vai tyhmä… vai molempia?” Perustelkaa pohdintojanne ja valmistautukaa kertomaan joku ajatuksistanne muulle luokalle.

Keskustelun koonti (10min)
Jokainen pari kertoo muille jalostuneimman ajatuksensa. Opettaja nostaa esiin erityisesti ne kommentit, joissa on huomioitu ihmisen vaikutus koneen toimintaan. Lisätietoa tästä aiheesta ja kerrottavaa oppilaille löytyy oivallisesti materiaalista: Ohjelmointiopas alakouluun, s.9. Ohjelmointia_alakouluun_SYK_2017.pdf

Operobotti (10min)
Opettaja istuu tuolilla. Oppilaiden tehtävänä on ohjelmoida suullisin ohjein opettaja kävelemään ovelle ja avaamaan ovi. Yksi oppilaista toimii puheenvuorojen jakajana. Ohjelmoinnin jälkeen keskustellaan mm.

- käskyjen täsmällisyydestä ja oikeasta suoritusjärjestyksestä
- voiko vaikkapa käskyn ”astu yksi askel eteenpäin” korvata käskyllä ”astu 5 askelta eteen päin”?
- oliko käskyissä joitain toistuvia osia? (silmukka)
- tällaista yksityiskohtaista toimintaohjeiden sarjaa kutsutaan algoritmiksi (ope kirjoittaa sanan tauluun)

Parin ohjelmointi (15min)
”Sopikaa, kumpi on ensin ohjelmoija ja kumpi robotti.
1) Ohjelmoi robotti ottamaan englannin kirja repusta pulpetille.
Vaihtakaa osia.
2) Ohjelmoi robotti piirtämään paperille neliö matematiikan vihkoon. Lähtötilanteessa vihko ja kynä ovat pulpetilla.
Lisätehtävänä voit keksiä oman ohjelmointitehtävän.”

Opettaja seuraa parien toimintaa ja antaa palautetta onnistumisista sekä ohjaavin ja aktivoivin kysymyksin auttaa pulmakohdissa eteen päin.

Tehtävän lopussa vertaisarviointi: ”Kerro parillesi, mikä hänellä onnistui ohjeiden/käskyjen antamisessa erityisen hyvin.”

Kootaan opittu (5min)
Ensin 1-2min parin kanssa keskustellen: ”Mitä tarkoittaa algoritmi?” Parikeskustelun jälkeen kuunnellaan muutaman parin selitys ja opettaja täydentää tarvittaessa:

- Algoritmi tarkoittaa käskyjonoa. Tietokoneet osaavat toimia ihmisten laatimien algoritmien avulla. Algoritmeja voidaan nähdä ja käyttää myös monissa arjen tilanteissa (hampaiden pesu, leivonta, koneet, …)

Peukku pystyyn -arviointi (1min):

- Näyttäkää peukulla (ylös/sivulle/alas),
    - kuinka hyvin teidän parinne ymmärsi ja osasi selittää käsitteen algoritmi
    - kuinka tasa-arvoisesti jaoitte ajatuksianne keskustelutehtävissä
    - kuinka hyvin keskityitte tämän tunnin toimintatehtäviin?

Videot, jos aikaa jää (5min)
Katsotaan kaksi lyhyttä videota materiaalista Linda Liukas: Rakkauskirjeitä tietokoneelle:

- Video 3 ”Tietokoneen mahdollisuudet”
- Video 4 ”Kaikenlaiset koodarit”

Tavoitteena on videoiden avulla laajentaa oppilaiden ajattelua ja käsitystä ohjelmoinnin mahdollisuuksista yhteiskunnassa.

TUNTI 2: Scratchin alkeita

Oppilailla on jokaisella oma tietokone, mutta työskennellään edelleen parin kanssa.

Scratchiin tutustuminen (15min)
Kirjaudutaan Scratch-ohjelmaan. Opettaja esittelee ohjelmointiympäristöä samalla, kun oppilaat omilla koneillaan tutkivat ja kokeilevat opettajan esittelemiä asioita. Käydään läpi ohjelmointinäkymä, hahmojen ja taustojen luominen sekä ennen kaikkea työkalupakki. Katso hyvät vinkit Pulmaario-oppaasta s. 16-20
Pulmaario_ohjaajanopas.pdf (helsinki.fi)

Tehtävä (25min)
Animoidaan oman nimen kirjaimet liikkumaan hiirtä klikkaamalla.
Tästä löytyy hyvä ohjevideo Scratchin oppitunnit-osiosta.
Tavoitteena on ymmärtää, että jokainen hahmo tarvitsee oman käskysarjan toimiakseen ja kokeilla erilaisia toimintoja.

1) Opettajan ohjeiden mukaisesti yhtä aikaa luodaan omalle animaatiolle tausta sekä haetaan oman nimen kirjaimet hahmoiksi.
2) Yhdessä tarkastellaan ensimmäisen kirjaimen ohjelmointi. Valitaan toiminto ”kun tätä hahmoa klikataan” ja sen jälkeen lisätään ääni. Kokeillaan vihreää lippua painamalla, toimiiko koodi.
3) Tämän jälkeen oppilaat itse valitsevat ja kokeilevat muille kirjaimille toimintoja, joita hiiren klikkauksesta seuraa. Näitä voivat olla vaikkapa värin tai koon vaihtuminen tai kirjaimen liikkuminen tai pyörähtäminen.

Opettaja kiertää ohjaamassa oppilaita ja auttamassa tarpeen mukaan. Ensisijaisesti opettaja rohkaisee kokeilemaan eri komentoja, ja kysymyksin ohjaa löytämään toivotut vaihtoehdot tai pohtimaan, miksi koodi ei kenties toimi toivotusti. Työskentelyssä kannustetaan kaverin neuvomiseen ja auttamiseen.

Loppukoonti (5min): ”Nosta käsi ylös
- jos sait kaverilta hyvän vinkin,
- jos autoit kaveria,
- jos sinulla pulma ohjelmoinnissa,
- jos jaksoit pulmasta huolimatta yrittää ja jatkaa eteen päin,
- jos keksit ratkaisun omaan tai kaverin pulmaan,
- jos ehdit ohjelmoida puolet tai enemmän nimesi kirjaimista.”

Seuraavilla oppitunneilla jatketaan Scratchin avulla ohjelmointia ja edetään monimutkaisempiin käskysarjoihin, silmukoihin jne. Nyt kun oppilailla on käsitys siitä, mitä algoritmi tarkoittaa, voidaan algoritmeja nostaa esiin arjen tilanteissa vaikkapa internetin käytöstä tai koneiden toiminnasta keskustellessa ja näin lisätä heidän tietoisuuttaan matematiikan olemassaolosta ”kaikkialla”.

Lähteet:

Konttinen, M. 2017. Ohjelmointiopas alakouluun. 20 oppituntia 3-6-luokkalaisille. Opettajalle tarkat ohjeet oppitunneille. Ohjelmointia_alakouluun_SYK_2017.pdf

Liukas, L. Rakkauskirjeitä tietokoneelle. Otavan opepalvelun sähköiset materiaalit.

Pulmaario_ohjaajanopas.pdf (helsinki.fi)

Tarinan tekeminen Scratchilla, 5. lk

Tiina Niiranen, CC BY-SA 4.0

9.11.2021

SCRATCHIIN TUTUSTUMINEN JA TARINAN TEKEMINEN SCRATCHILLA

Tuntisuunnitelman tekijä: Tiina Niiranen

Suunnittelin kahden tunnin kokonaisuuden, jossa 5. luokkani oppilaat tutustuvat Scratchin käyttöön ja tekevät sillä parin kanssa pienet tarinat. Aiheena on: “Yllättävä sattumus”.

Tarinoiden tekemiseen liittyy osalla oppilaista kielteisiä mielikuvia: “En mä kuitenkaan keksi mitään!” tai “En osaa kuitenkaan kirjoittaa hyvää tarinaa!” Tämän tehtävän avulla tarinan keksimisen kynnystä on madallettu nopan heiton myötä ja kirjoittamisen tilalle on tuotu animaation tekeminen.

ScratchJr on luokalleni tuttu sovellus ja sillä olemmekin tehneet useita tarinoita ja animaatioita. Ennen näitä tunteja olisi tarkoitus luoda oppilaille omat Scratch-tunnukset. Siihen kannattaa varata aikaa, koska oppilaiden vanhemmilta tulee kysyä lupa (tunnuksen luominen vaatii vanhemman sähköpostiosoitteen).

Oppituntien tavoitteet:

- Äidinkieli:
    - Lyhyen tarinan juonen keksiminen.
    - Yllättävän huippukohdan keksiminen tarinalle.
    - Vuoropuhelun keksiminen.
- Ohjelmointi:
    - Scratch-ohjelmaan tutustuminen
        - Hahmot
        - Taustat
        - Aloittaminen ja koodipalikoiden liittäminen toisiinsa
        - Liike
        - Puhekuplat
        - Taustan vaihtuminen
- Työskentely:
    - Parin kuunteleminen.
    - Omien ideoiden kertominen rohkeasti parille.

Tarvittavat laitteet/välineet:

- oppilaille laitteita, joilla Scratch toimii (tietokone/läppäri/Chromebook), laite per pari
- opelle tietokone/läppäri tms. sekä laitteisto, jolla voi näyttää joko oman tietokoneen näyttöä tai valmista opetusvideota taululle
- parinarvontakone internetissä: https://wheelofnames.com/
- noppia
- Liitteenä olevat taulukot joko monistettuna tai sähköisinä heijastettaviksi oppilaille.

Oppituntien kulku

Motivointi *5 min*

Johdatus aiheeseen, jossa opettajan oma innostus on erittäin tärkeää:

“Kuvittele, jos ritari tapaisi dinosauruksen jäätiköllä! Mitä ihmettä? Miksi he siellä ovat? Mitä he sanoisivat toisilleen? Mitä he siellä tekisivät? Ja mitä ihmettä puolestaan on tapahtunut, kun he kohta ovatkin keskellä koripallokenttää? Mikä yllättävä käänne heidät sinne sai?”

Taustalla voi toki olla esim. kuvat ritarista ja dinosauruksesta jäätiköllä, tai jättää ne täysin oppilaiden mielikuvituksen varaan.

Tehtävänanto *10 min*

“Tänään pääsette tekemään parin kanssa tarinan, jossa on jotain yllättävää. Parikin on tänään yllätys, sillä parit arvotaan!”

(https://wheelofnames.com/ Jos et ole käyttänyt aiemmin, käy kirjoittamassa etukäteen luokkasi oppilaiden nimet, niin siihen ei mene tunnilla aikaa.)

Parien arvonnan jälkeen opettaja antaa nopat oppilaille, sillä tarinan päähenkilöt ja tapahtumapaikatkin ovat vielä yllätyksiä ja ne arvotaan siis nopalla liitteenä olevan taulukon mukaisesti (taulukko 1.)

Kerrotaan, että tehdään tarina parin kanssa Scratchillä.

Kerrotaan tavoitteet tunnille, tunnin kulku ja ajankäyttö. Kerrotaan, että tällä kertaa tarina on lyhyt ja ytimekäs. Olennaista on keksiä

- alkutilanne: Mitä hahmot tekevät/sanovat?
- huippukohta: Mitä ihmettä tapahtuu? Miten he joutuvat toiseen paikkaan?
- lopputilanne: Mitä hahmot tekevät/sanovat?

Scratchin käytön ohjeistus *10-15 min*

Opettaja voi näyttää Scratchin käytön perusasiat itse, jos osaa.

- sisäänkirjautuminen
- uuden projektin luominen
- hahmojen luominen
- taustojen valinta ja vaihtuminen
- eri koodiryhmien esittely lyhyesti (lähinnä liike ja puhuminen)
- oman ohjelmoidun jutun testaaminen, toimiiko se

Jos opettaja ei tunne Scratchiä, voidaan katsoa yhdessä opetusvideo, esim. “Scratch-opetusvideo aloittelijoille” https://www.youtube.com/watch?v=FTAoABcmph (Huom! Scratch-opetusvideot ovat joitakin vuosia vanhoja, joten ulkoasu on hieman muuttunut.)

Tarinan tekeminen Scratchillä *45 min*

Annetaan oppilaiden tehdä omaa tarinaansa. Annetaan ohjeeksi, että parien täytyy vaihdella koodaajaa tunnin aikana useita kertoja, jotta molemmat pääsevät kokeilemaan ohjelmoimista.

Tässä vaiheessa opettajan tehtävänä on havainnoida oppilaiden työskentelyä ja antaa heti palautetta oppilaille oppituntien tavoitteeseen nähden. Kiinnitetään huomiota siis tarinan keksimiseen, itse ohjelmointiin sekä parin kanssa työskentelyyn.

Kannustetaan kokeilemaan ja kysymään myös toisilta pareilta neuvoa.

Opettajan tehtävänä on myös pitää oppilaat tekemässä olennaisia asioita, jotta tarina saadaan tehtyä loppuun. (Ainahan tarinaa voi näiden tuntien jälkeen jatkojalostaa ja keksiä siihen lisää hienouksia!)

Vertaispalaute *5-10 min*

Vaihdetaan laitteita jonkun toisen parin kanssa. Parit kertovat toisilleen, mitä nappeja painamalla tarina toimii. Katsotaan parin tarina.

Annetaan palaute parille peukkua näyttämällä (kysymykset näkyvissä esim. taululla). Eli ensin toinen pari arvioi toisen tarinan: molemmat oppilaat näyttävät peukkua ylös, alas tai johonkin siltä väliltä (esim. Oliko tarinan huippukohta yllättävä?). Lisäksi he kertovat lyhyesti, mikä tarinassa oli parasta. Sitten sama toisinpäin.

Itsearviointi *5 min*

Jokainen oppilas arvioi omaa työskentelyään. Opettaja kysyy liitteenä (taulukko 3.) olevat kysymykset oppilailta, ja oppilas näyttää peukulla ylös, alas tai johonkin siltä väliltä.

Liitteet

Taulukko 1. Päähenkilöiden ja tapahtumapaikkojen arpominen.

Nopan silmäluku	Ensimmäinen päähenkilö	Toinen päähenkilö	Ensimmäinen tapahtumapaikka	Toinen tapahtumapaikka
1	dinosaurus	ritari	jäätikkö	pelikenttä
2	lepakko	poro	avaruus	aavikko
3	noita	lumiukko	ranta	makuuhuone
4	robotti	lohikäärme	meri	metsä
5	sukeltaja	luuranko	noidan talo	konsertti
6	hai	norsu	linna	kaupunki

Taulukko 2. Vertaispalautteen antaminen.

Näytä peukulla arviosi näkemästäsi tarinasta:
1. Oliko juoni selkeä? Ymmärsitkö, mitä tarinassa tapahtui?
2. Oliko tarinan huippukohta yllättävä?
3. Toimiko tarina saamienne ohjeiden mukaan (vai oliko siihen jäänyt bugeja eli ohjelmointivirheitä)?
Kertokaa vielä lyhyesti, mikä oli parasta näkemässänne tarinassa.

Taulukko 3. Itsearvioinnin antaminen.

Näytä peukulla arvio omasta työskentelystäsi:
1. Kuuntelitko pariasi?
2. Ehdotitko rohkeasti omia ideoitasi?
3. Osaisitko tehdä Scratchillä uuden tarinan, jossa käytät näitä perustaitoja?

Kuplalajittelu, 5. lk

15.10.2022

Sain idean tuntisuunnitelmaani eräältä toiselta kurssilaiselta, joka esitteli aiemmin lukemassani vastauksessaan kuplalalajittelualgoritmia.

Tunnin tavoitteena on harjoitella kuplalajittelua ensin yhdessä, sitten itsenäisesti parityönä ja lopuksi kirjoittaa sanallinen koodi, jonka avulla tällaisen lajittelun voi suorittaa. Samalla tavoitteena on herättää keskustelua siitä, miten ihmisen ja koneen tekemät ratkaisut eroavat toisistaan.

Opettaja arvioi oppitunnin aikana oppilaiden parityöskentelyä sekä heidän tuottamaansa kirjallista koodia/ohjetta. Oppilaat arvioivat omaa ja toistensa osaamista testaamalla kirjallista ohjeistusta.

Suunnittelen tunnin omalle viidennelle luokalleni, mutta tehtävä on sen verran helppo, että se onnistuu varmasti myös pienempien kanssa.

Oppitunnilla tarvitaan lukukortteja oppilasmäärän verran, jokaiselle parille omat numerokortit 0-9 sekä muistiinpanovälineet.

Tunnin ensimmäisessä tehtävässä oppilaat menevät seisomaan riviin ja opettaja jakaa jokaiselle oppilaalle kortin, joissa on sattumanvarainen luku. Selkeyden vuoksi kaikilla oppilailla on kortissaan eri luku. Tavoitteena on saada järjestettyä oppilaat lukukortteineen niin, että ensimmäisenä vasemmalla on pienin luku ja oikealla puolestaan suurin. Lämmittelyvaiheessa oppilaat saavat järjestäytyä riviin vapaavalintaisella tyylillä, kunhan lopputulos on oikea.

Opettaja nostaa esille, että oppilaat osasivat ajatella luovasti ja järjestäytyä riviin oikealla tavalla. Tietokone ei kuitenkaan osaa ajatella luovasti, vaan noudattaa sille annettuja käskyjä. Yksi tapa järjestää luvut on kuplalajittelu, jossa kone käy rivin läpi verraten aina kahta vierekkäistä lukua ja vaihtaa niiden paikkaa, jos järjestys on väärä. Kone käy riviä läpi niin monta kertaa, että vaihdettavia pareja ei enää löydy. Toistetaan sama harjoitus niin, että oppilaat saavat korttinsa uudelleen ja tehdään sattumanvarainen rivi. Nyt opettaja aloittaa reunasta ja vaihtaa oppilaiden paikkaa pareittain niin kauan, että saadaan oikea järjestys.

Kokeillaan kuplalajittelua parin kanssa. Parit saavat pöydälleen matematiikan kirjan numerokortit 0-9. Asetetaan numerokortit sattumanvaraiseen järjestykseen ja järjestetään suuruusjärjestykseen pari kerrallaan vaihtamalla.

Oppilaat laativat pareittain kirjallisen ohjeen, jonka mukaan toimien tietokone tai robotti osaisi suorittaa saman lajittelun.

Parit testaavat toistensa ohjeita ja antavat palautetta siitä, voiko sen mukaan suorittaa lajittelun oikein. Tässä vaiheessa kiinnitetään erityistä huomiota siihen, että ohjeen mukaan toimitaan täsmällisesti, vaikka se ei tuottaisi haluttua lopputulosta. Omaa koodia on mahdollista muokata testaamisen jälkeen.
https://fi.wikipedia.org/wiki/Kuplalajittelu

Animaation tekeminen Scratchilla, 5. lk

13.11.2022

Taustatietoa: Tämän tuntisuunnitelman toteuttamiseksi opettajalla ja oppilailla täytyy olla aiempaa kokemusta Scratchin käytöstä sekä omat tunnukset Scratch -ohjelmointiympäristöön. Oma luokkani on tehnyt aiemmin kurssilta tuttuja keräilypeli -tyyppisiä pelejä. He ovat myös tehneet animaatioita Stop motion ohjelman avulla, joten animaatiot itsessään ovat myös hyvin tuttuja heille.

Kokonaisuus voidaan toteuttaa joko pari- tai yksilötyönä.

Tuntisuunnitelma

Aihe: Animaatio, ohjelmointiympäristönä Scratch. Aihe voidaan sisällyttää lähes kaikkien oppiaineiden sisältöihin aiheesta riippuen. Tämä tuntisuunnitelma on suunniteltu pidettäväksi matematiikan tunnilla.

Tavoitteet: Oppilas syventää ohjelmointitaitojaan. Oppilas tutustuu hahmon animointiin Scratch -ohjelmointiympäristössä. Oppilas osaa luoda lyhyen animaation Scratchilla.

Sisällöt: Oppilas suunnittelee ja toteuttaa animaation graafisessa ohjelmointiympäristössä (Scratsch). Aikaa 2 x 45 min.

Arviointi: Arviointi perustuu jatkuvaan havainnointiin. Opettaja kyselee, kannustaa ja kommentoi tunnin edetessä. Oman työn valmiiksi saattamisen jälkeen työ jaetaan muiden tarkasteltavaksi. Jokainen oppilas tutustuu vähintään kolmen luokkatoverin/parin tekemään animaatioon ja antaa lyhyen sanallisen palautteen joko suoraan Scratchin kommenttikenttään tai suullisesti.

Ohjelmointiympäristö antaa välitöntä palautetta ohjelmoinnin onnistumisesta. Tätä palautetta oppilaat voivat hyödyntää osana itsearviointia. Tunnin lopuksi opettaja pyytää itsearvion oman projektin onnistumisesta ja sen mielekkyydestä esimerkiksi nopealla peukkuarvioinnilla.

Työskentelyvälineet: Cromebook jokaiselle oppilaalle/parille, lisäksi tarvitaan nettiyhteys

Tunnin kulku:

Jokainen oppilas/pari ottavat Cromebookit ja kirjautuvat sisään. Oppilaat kirjautuvat omilla/toisen pareista tunniksilla Scratch -ohjelmointiympäristöön. Kerrataan yhdessä pikaisesti Scratchin käyttöä.

Luodaan uusi projekti. Oppilaita ohjeistetaan lähtemään liikkeelle valitsemallaan animaatiolle tausta ja taustaan sopiva hahmo/hahmot. Omasta innostuksesta ja taitotasostaan riippuen jokainen lähtee ohjelmoimaan omaa hahmoaan lausekkeiden avulla. Animaatiossa tulee olla ääntä ja liikettä. Etevimmät voivat luoda pidemmän

animaation, jossa on selkeästi nähtävissä juonellinen tarina. Oppilaita on hyvä muistuttaa kokeilemaan välillä koodin toimivuutta. Oppilaita tulee rohkaista auttamaan luokkatovereita ongelmakohtien ratkaisemisessa. Oman työn valmiiksi saattamisen jälkeen työ jaetaan muiden tarkasteltavaksi.

Jokainen oppilas tutustuu vähintään kolmen luokkatoverin/parin tekemään animaatioon ja antaa lyhyen sanallisen palautteen joko suoraan Scratchin kommenttikenttään tai suullisesti. Mikäli tuplatunnin loppuun jää aikaa, animaatioita voidaan tarkastella yhdessä taululle heijastettuna.

Pelin ohjelmointi Scratchilla, 5.–6. lk

Veera Salminen, CC BY-SA 4.0

23.10.2020

Pelin ohjelmointia Scracthillä

Kokonaisuudessa harjoitellaan 5. – 6.-luokkalaisten kanssa pelisuunnittelua ja ohjelmointia Scratch:n avulla. Tavoitteena on luoda pienimuotoinen peli Scratch-ohjelmalla. Osoite: https://scratch.mit.edu/

Aiempi osaaminen ja motivointi:

Oppilaiden kanssa on jo tutustuttu Scratch-ohjelmointiympäristöön eli oppilaat eivät käytä sitä ensimmäistä kertaa. Myös koodien rakentaminen on tuttua yhdellä hahmolla.

Tällöin motivointina voi helposti toimia aiempaan ohjelmointihetkeen palaaminen. Projektia voi käyttää myös osana esimerkiksi kummioppilas-toimintaa, jossa oppilaat tekevät pelit kummioppilaidensa tai vastaavasti vain nuorempien oppilaiden pelattavaksi.

Oppimiskokonaisuuden tavoitteet:

Kokonaisuuden tavoitteet Perusopetuksen opetussuunnitelman (2014) mukaan:

- T1: Ylläpitää innostusta matematiikkaa kohtaan ja tukea oppilaan itseluottamista
- T5: ongelmaratkaisutaitojen kehittäminen
- T14: innostaa oppilasta laatimaan toimintaohjeita tietokoneohjelmilla

Lisäksi projektikohtaiset tavoitteet:

- Luodaan toimiva pienimuotoinen peli (esim. parin minuutin mittainen)
- Harjoitellaan pelin ohjelmointia vähintään kahdella hahmolla
- Harjoitellaan hyödyntämään pelissä piste- ja/tai aikamuuttujaa

Työskentelyn tavoitteet:

- Harjoitellaan ongelmanratkaisutaitoja
  - Yritän ensin itse ja ongelmatilanteessa pysähdyn miettimään tilannetta rauhassa.
  - Autan kaveria keskustelemalla ongelmatilanteesta ja annan jokaiselle rauhan ohjelmoida itse vinkkien avulla.

Opettajan arviointi:

Opettaja arvioi oppilaan työskentelyä projektin aikana. Painopisteenä ongelmaratkaisutaitojen kehittäminen ja arviointi on formatiivista eli ohjaavaa. Opettaja voi työskentelyn aikana kysyä oppilaalta, kuinka suunnittelu on edennyt ja miten ongelmatilanteet ovat ratkenneet. Keskustelussa opettaja voi palata työskentelylle asetettuihin tavoitteisiin: ”yritän ensin itse ja pysähdyn miettimään” tai ”autan kaveria keskustelemalla ongelmatilanteesta”.

Lopputuotoksen arviointi: Opettaja voi peilata lopputulosta oppilaan suunnitelmaan.

- Onko peli pelattavissa?
- Onko pelissä vaaditut elementit: vähintään kaksi hahmoa sekä piste- tai aikamuuttuja?

Oppilaan itsearviointi:

(Lomake liitteenä)

Tarvittavat välineet:

Oppikokonaisuudessa tarvitaan muistiinpanovälineet sekä Scratch:n käyttöön sopiva laite. Lisäksi itsearviointiin tulostetut lomakkeet tai luokan käytänteiden mukaan esimerkiksi opettajan jakama word-tiedosto. Halutessaan voi käyttää suunnittelussa valmista pohjaa (liitteenä).

Projektin ohjeistus:

Tarkoituksena suunnitella ja toteuttaa pienimuotoinen peli, jossa harjoitellaan useamman kuin yhden hahmon sekä muuttujan käyttämistä. Esitellään tavoitteet. Edellä mainitut tavoitteet on hyvä näyttää oppilaille sekä avata niitä sanallisesti.

Projektissa työskentelyssä painotetaan ongelmaratkaisutaitojen kehittämistä. Ongelmatilanteita yritetään ratkoa itse pysähtymällä hetkeksi ja rauhassa etsimään ratkaisuvaihtoehtoja. Myös kaverin kanssa

keskustelu auttaa eteenpäin. Annetaan kuitenkin jokaiselle mahdollisuus ohjelmoida itse.

Projektin alussa on hyvä kerrata ohjelmoinnin perussanastoa ja se, kuinka Scratch toimii. Tärkeimpiä elementtejä:

- Kuinka hahmo luodaan
- Miten hahmoa liikutetaan
- Miten erilaiset koodit muodostetaan (Liike, ulkonäkö, ääni, tapahtumat, ohjaus, tuntoaisti, toiminnot, muuttujat,lohkoni)
- Lisäksi käydään yhdessä esimerkin omaisesti, kuinka muuttujia käytetään. (Voidaan käyttää myös valmista peliä, jonka koodeja tarkastellaan yhdessä)

Esimerkki peli, jota voi tutkailla yhdessä oppilaiden kanssa: ”Pöllön yhteenlaskua luku-alueella 1-30”
https://scratch.mit.edu/projects/428886400 Tekijä: Veeps_ (24.9.2020).

Projektin aikataulu:

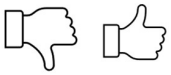
Käytettävissä on 2 oppituntia (2x45min)

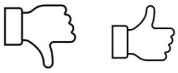
1. Tunnin aikana	Tehtävä	Käytettävissä oleva aika (suuntaa antavia)
	Tehtävänanto, aiemman kertaus	15min
	Muistiinpanojen tekeminen	5min
	Ohjelmoinnin aloitus	25min
2. Tunnin aikana	Ohjelmoinnin jatkamista	n. 30min
	Itsearviointi	10min
	Loppukoonti	2min

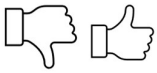
Loppukoonnissa voidaan pohtia jonkin oppilaille tutun pelin rakennetta, millaisia muuttujia tai koodeja siinä voisi olla. Näin voidaan liittää opittu oppilaiden kokemusmaailmaan.

Projektin itsearviointi:

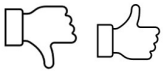
Nimi:_____

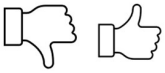
Onnistuin pelin suunnittelussa ja toteutuksessa: 

Ohjelmointi oli mielestäni hauskaa: 

Onnistuiko pelin pelaaminen, kuten suunnittelit? 

Onko pelissä vaaditut elementit:

Vähintään kaksi hahmoa 

Piste- tai aikamuuttuja 

Minkälaisissa tilanteissa pysähdyit miettimään, kuinka pääset eteenpäin tai kuinka saat tietyn elementin koodattua oikein:

Mikä auttoi sinua eteenpäin kyseisessä tilanteessa:

Miten autoit kaveria hänen projektissaan:

Suunnittelun avuksi:

Mitä hahmoja pelissä on (vähintään kaksi):

Mitä hahmot tekevät? Keräävätkö jotain, liikkuvatko karkuun?

Mitä muuttujaa pelissä käytät ja miten? Keräävätkö hahmot pisteitä tai liikkuvatko aikaa vastaan?

Kirjoita lyhyesti, mikä on pelin tarkoitus?

Jos ohjelmoinnissa jää aikaa, voit miettiä tarkemmin hahmojen ulkonäköä ja pelitaustaa.

Ohjelmoinnin ensiapua opettajalle:

Scratchiin tarvitaan tunnukset, jotka voi luoda helposti sähköpostin avulla. Mikäli Scratchin toimintaperiaate on hukassa, ohjelmiston sivujen on ”oppitunnit”- kokonaisuuksia, jossa on videoina ohjeita ja vinkkejä erilaisten tuotosten luomiseen. Myös Youtubessa sekä LUMA-keskus suomi materiaaleissa on runsaasti vinkkejä.

Muuttujien ja hahmojen käyttäminen

Esimerkki pelissä on kolme hahmoa: pöllö ja kaksi omenaa. Muuttujia pelissä on kuusi: aika ja pisteet sekä yhteenlaskuissa käytettävät luvut.

Pelaajalla on 10s aikaa kerätä omena. Kosketettuaan omenaa, ruudulle tulee yhteenlaskutehtävä. Vastattuaan oikein, pelaaja saa pisteitä. Jälleen aikaa 10s kerätä uusi omena.

Näin peli jatkuu, kunnes pelaaja saa 20 pistettä tai ei ehdi seuraavan omenan luo ajan kuluessa.

Malli-pelissä on siis huomattavasti enemmän elementtejä, kuin projektin tavoitteissa. Toisaalta tämä antaa mahdollisuuden eriyttää ylöspäin ja siten näyttää oppilaille, mihin muuttujia käytetään ja minkälaisia muuttujia voi olla.

Selitys siihen, miksi hahmoja on kolme, vaikka omenat ovat samanlaisia, löytyy yhteenlaskutehtävistä. Omenoiden muuttujat yhteenlaskussa, toisella on ”eka” ja ”toka”, toisella omenalla ”kolmas” ja neljäs”. Lisäksi vastauksista tulee erilaiset kommentit.

Alla olevasta linkistä pääsee suoraan kurkkaamaan projektin sisälle ja tutkailemaan koodeja.

https://scratch.mit.edu/projects/428886400/editor/

Lähteet:

Opetushallitus. (2016). Perusopetuksen opetussuunnitelman perusteet. Helsinki: Next Print Oy.

Pöllön yhteenlaskua lukualueella 1-30. Tekijä: Veeps_ https://scratch.mit.edu/projects/428886400 Liitetty: 23.10.2020

Ohjelmoinnin kokonaisuus ScratcJr:lla ja Scratchilla, 5.-6. lk

13.11.2022

Suunnittelen kokonaisuuden 5.-6.lk:n valinnaisen tietotekniikan ryhmälle. Suunnittelin ohjelmoinnista itsenäisen kokonaisuuden, sillä oppilaiden toive on ollut oppia tekemään peli. Lähtötasoltaan oppilailla on eroa, pari on itsenäisesti harjoitellut ohjelmointia, mutta suurimmalle osalle ohjelmointi on täysin uusi asia.
Tavoitteena on luoda ymmärrys mitä ohjelmointi on, kuinka ohjelmointikieli toimii.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi:

Tavoitteena on luoda ymmärrystä, mitä ohjelmointi on ja kuinka Scratch-ohjelmointikieli toimii.

1. Tutustumme Scratch-ohjelmointikieleen lyhyiden videoiden ja ohjeiden avulla.

2. Jokainen harjoittelee Scratch jr-sovelluksella komentojen käyttöä ja tekee animaation, jossa on ainakin kaksi hahmoa ja vähintään kaksi ruutua.

3. Teemme oppilastunnukset Scratch-sivustolle ja harjoittelemme pelin yksinkertaisen pelin ohjelmointia.

Koulun Ipadeilla on ladattuna Scratch jr-sovellus ja olen luonut jo Scratch-sivustolle opettajatunnuksen.

Arviointi:

- -Tavoitteena on edetä rauhassa vaiheittain, edeten yksilöllisesti aina vaikeampiin toimintoihin. Tarkastelen oppilaan oman ohjelmointitaidon yksilöllistä kehittymistä ja algoritmista ajattelua.
- - Arvioitavana on oppilaan työskentely, ohjelman mahdollisuuksien ymmärtäminen ja hyödyntäminen sekä ohjelmointitaidon kehittyminen ja keskittyminen omaan työskentelyyn.
- - Opettajan formatiivinen arviointi ja palaute työn edetessä. Kannustaen eteenpäin. Animaatiot esitetään ryhmälle ja annetaan vertaispalautetta. Pyritään nostamaan esiin onnistumiset ja kunkin työn saavutukset.

Työskentelyvälineet ja opetusmenetelmät

- - Työvälineinä käytetään sekä Ipadeja ja Cromebookeja. Ohjelmistona Scratch jr ja Scratch
- - Johdatus aiheeseen: opettajan PowerPoint- jossa käydään läpi keskeisimmät Scratch-kielen piirteet ja komennot.
- - Video: https://opentunti.fi/plans/show/2231/koodauksen-perusteet-alakouluissa-scratch-jr
- - Classroom: ohjeet ja linkkien jakaminen tunnuksen tekemiseen Scratch-ympäristöön
- - Valitut Scratch-ohjelmointikortit tulostettuna.

Motivointi ja palaute

- Ohjelmointi lähtee oppilaiden toiveesta, joten virittäydymme aiheeseen tutustumalla muutamiin projekteihin, missä näkee mitä ohjelmalla on mahdollisuus saada aikaan. Kerron tunnin tavoitteen ja sen, että saamme lopuksi nähdä kaikkein animaatiot.
- Kiitän ja kannustan auttamisessa. Kun joku osaa ja oppii toiminnon, pyydän mieluummin oppilasta kertomaa sen toiselle. Pidän oman roolini ohjaavana ja pyrin auttamaan haastavissa kohdissa eteenpäin. Oppilaat, jotka ovat aiemmin ohjelmoineet, saavat vastuutehtäviä. He voivat edetä tavoitteissa monimutkaisempiin komentosarjoihin.
- Valinnaisryhmä voi toimia oman luokkansa tutorina ja näyttää mallia yhdistäessään opittuja taitoja muiden aineiden projekteja tehdessä.

Ohjelmointia kotitalouden opetukseen , 6.lk

Mari Muhonen, CC BY-SA 4.0

10.3.2021

Tuntisuunnitelma 6.lk valinnainen kotitalous ja ohjelmointi

"VIITTÄ VÄRIÄ PÄIVÄSSÄ"

Aihepiirin valinta ja rajaus

Haluan kokeilla soveltaa ohjelmointia kotitalouden opetukseen. Kotitalouden taidoissa on paljon algoritmista ajattelua vaativaa oppimista ja on kiinnostava kokeilla tuoda nämä kaksi käytännönläheistä mutta eri tavalla konkreettista ainetta yhteen. Kotitaloudesta aiheena on **kasvisten päivittäinen saantisuositus** ja ohjelmoinnin puolelta **algoritmisen ajattelun harjoittaminen** koodinkirjoitustehtävällä sekä jakson lisätehtävänä kasvispelin suunnittelu.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Kyseessä on kuudennen luokan valinnainen kotitalouden kurssi. Kotitalouden puolelta tavoitteena on **tutustua kasvisten päivittäiseen saantisuositukseen "viittä väriä päivässä"**. Ohjelmoinnin osuuden **tavoitteena on harjoittaa algoritmisen ajattelun taidon soveltamista arjen puuhassa.** Konkreettinen tavoite on saada kirjoitettua sellainen "koodi" eli vaiheittainen ohje, jota noudattamalla toinen pari saa hedelmän kuorittua. Lisätehtävänä on suunnitella aiheeseen liittyvä peli ja vahvistaa pelin koodaamisen taitoja Scratchillä.

Opettaja arvioi molempien tehtävien suorittamista lähinnä **formatiivisesti**. Pääpaino tässä suunnitelmassa on ryhmässä tapahtuvalla **vertaisarvioinnilla ja- palautteella**. Tehtävä myös ikäänkuin arvioi itse itseään, sillä mikäli kirjoitettu" kuorintakoodi" ei toimi, niin hedelmää ei saa kuorituksi.

Työskentelyvälineet ja opetusmenetelmät

Välineiksi oppikokonaisuuteen tarvitaan: Kotitalousluokka, hedelmiä ja/tai kasviksia, joilla harjoitus tehdään, opettajan kone ja tykki/tms., oppilaille kynät ja paperia sekä lisätehtävää varten chromebookit/padit/tms. *Lisätehtävävää varten opettajalle ja oppilaille Scratchissä työskentely ja keräilypelin/tms. tekemisen on oltava jonkin verran tuttua.*

Johdantona ja lämmittelynä **luokitellaan yhdessä ryhmänä** kasviksia väreittäin. Tämän jälkeen kerrataan kasvisten sukupuu **opettajajohtoisesti**.

Tunnin päätehtävänä oppilaat tekevät **pareittain** tarkat, vaiheittaiset ohjeet, jolla omenan/porkkanan/omavalintaisen kasviksen kuoriminen onnistuisi (Algoritmin muodostaminen). Ohjeita testataan toisen parin kanssa **ryhmätyönä**, saadaanko hedelmä kuorittua?

Opetuskerta on suunniteltu parityöskentelyksi, joka jo työtapana rohkaisee ja kannustaa yhteistyöhön. Oppilaiden aiempi ohjelmointiosaaminen huomioidaan oppilaiden omien ideoiden kuuntelemisena ja kannustamisena siihen, että he käyttävät tehtävässä kaikkia aiheeseen liittyviä taitojaan.

Opitun asian, tässä tapauksessa algoritminen ajattelu arjen tilanteessa, siirtymistä muihin konteksteihin tuetaan jatkossa tunneilla ohjaavilla apukysymyksillä (Miten voisit pilkkoa tämän osiin? Mitä sellaista tapahtui, josta tämä tapahtuma johtui?...)

Opetuskerran rakenne (1 x 90 min opetuskerta)

Tunnin aloitus, läsnäolijat ja päivän aihe (n. 5 min)

Lyhyt johdanto aiheeseen ja tehtävä (n. 20 min.)

- Pareihin jako (kotiryhmät puolitetaan)
- Nimetään kahdessa minuutissa mahdollisimman monta eri kasvista viidessä eri värissä (punainen, oranssi, keltainen/valkoinen, vihreä ja sininen/liila) yhteisesti taululle
- Kerrataan opettajajohtoisesti kasvisten sukupuu ja käydään läpi kasvisten saantisuositus “viittä väriä päivässä”
- Kerrotaan tehtävän ohjeet ja aikaraja 5 min.

  Tehtävä: Valitkaa yksi kasvis, jonka kuorimiseen kirjoitatte mahdollisimman yksityiskohtaisen ohjeen eli “koodin”. Kuorija ei osaa tehdä muuta kuin noudattaa teidän koodianne.

Käydään läpi tehtäviä ja toteutetaan koodit luokassa “oppilasroboteilla” (n. 45 min)

- Oppilaspari vaihtaa kirjottamansa koodin toisen parin kanssa.
- Parista toinen lukee koodia ja toinen suorittaa. Kirjoittaja pari seuraa vierestä ja pattitilanteessa korjaa koodiaan toimivammaksi.
- Loppukeskustelu: Mitä tapahtui, kun koodi heräsi eloon? Olivatko vaiheet ja käskyt oikeassa järjestyksessä? Millaiset käskyt toimivat? Entä millaiset eivät? Millaisia muutoksia täytyisi palata tekemään?

Syödään kuoritut kasvikset välipalana ja siivotaan luokka (n. 15 min)

Lisätehtävän ohjeistus: Kasvispelin suunnittelu Scratchillä (loppuaika)

Huom! Opettajalla ja ryhmällä täytyy olla jo jonkun verran kokemusta Scratch-ympäristössä toimimisesta.

> Tehtävä: Suunnitelkaa Scratchillä tehtävä peli, jossa sovelletaan tunnilla aiemmin opittua. Peli voi olla esim. keräilypeli, jossa kerätään kasviksia. Miettikää peliin joku “koukku” tai taustatarina, joka houkuttelee luokkakaverin kokeilemaan sitä. Peliä ei ole tarkoitus saada valmiiksi nyt, vaan sitä voidaan jatkaa pitkin jaksoa. Peliä voi vielä halutessaan jatkaa ja hioa omalla ajalla.

Historian opetus ja algoritmisen ajattelun kehittäminen, 6. lk

3.3.2022

Tuntisuunnitelma algoritmisen ajattelun kehittäminen - 6lk renessanssista 1800-luvulle

Tunti on suunniteltu luokalle, jolla on hieman aikaisempaa kokemusta scratch ohjelman käytöstä ja sillä ohjelmoinnista. Olemme luokan kanssa aiemmin yhdessä ohjelmoineet tavallisen keräilypelin lasten itse valitsemilla taustoilla ja hahmoilla. Nohevimmat lisäsivät työhönsä myös vihollisia ja ääniefektejä, joten tämän tyyppinen tehtävänanto on heille jo hieman tuttu, toki ensimmäisellä kerralla ohjausta tarvittiin paljon enemmän. Aikaa kannattaa varata ainakin 2x45min, ellei jopa enemmän, jotta aikaa jää myös testata toisten pelejä.

Aihepiirin valinta ja rajaus

Valitsin aihepiiriksi historian renessanssista 1800-luvulle. Oppilaiden tehtävänä on siis joko valita itse haluamansa aihe tuolta ajanjaksolta tai sitten ohjelmoida peli opettajan antamaan aiheeseen. Aiheita voisi olla esimerkiksi:

- Kolumbus matkalla Amerikkaan: laiva pitää saada meren yli turvallisesti, sinne voisi ohjelmoida merihirviöitä ja myrskyjä estämään kulkua (labyrintti)
- Aurinkokuningas ottamassa kiinni herkkuja ja väistelemässä esimerkiksi kiviä (väistelypeli)
- Rokote estämässä virusta pääsemästä ihmisen kehoon (pong-maalivahti)
- Veronkerääjä keräämässä kultasäkkejä (matopelityyppisesti)

Oppilaat keksivät varmasti noin laajalta ajanjaksolta hauskoja ja varmasti spesifejäkin pelejä, joissa korostuu jokin pieni yksityiskohta. Tavoitteena on siis luoda visuaalinen toteutus jostain historian yksittäisestä tapahtumasta. Toki jos ryhmä on suuri ja aiempaa kokemusta ei paljon ole, voi lapset erimerkiksi jakaa ryhmiin ja antaa jokaiselle oman pelityypin: ryhmä A tekee labyrinttipelin, ryhmä B väistelypelin jne. Kaikille voi myös antaa samanlaisen pelityypin, jolloin jää vain aiheen miettiminen pelityyppiin sopivaksi. Näin opettajan on helpompi kontrolloida sitä, mitä kukakin tekee ja tarvittaessa auttaa ja ohjeistaa.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

- Kerrata ja vahvistaa opittua ohjelmointia ja siihen liittyviä käsitteitä
- Oppia uusia käskyjä ja komentoja scratchin maailmassa
- Käyttää luovuutta ja mielikuvitusta oman pelin suunnittelussa
- Saada varmuutta ohjelmointiin ja tietokoneen käsittelyyn
- Kerrata ja oppia historian asioista valitulla ajanjaksolla
- Tehdä yhteistyötä ja auttaa kaveria

Arviointi tapahtuu formatiivisesti opettajan ohjatessa oppilaita valitsemaan aihetta ja toteuttamaan oma työnsä. Työn voi tehdä halutessaan yksin tai parin kanssa, mutta opettaja seuraa jokaisen oppilaan osallistumista ja tekemistä. Työ arvioidaan kokonaisuutena, miten hyvin historialliset faktat on otettu huomioon ja miten hyvin

peli toimii. Tässä on myös sallittua ja suotavaa hyödyntää sekä opettajan että luokkakavereiden apua.

Lopuksi oppilaat arvioivat itse omaa peliään: perustuuko pelin idea historialliseen faktaan ja miten peli toimii. Oppilaat saavat myös pelata toistensa pelejä ja antaa niistä (positiivista ja rakentavaa) palautetta.

Työskentelyvälineet ja opetusmenetelmät

Oppitunnin aikana oppilaat tarvitsevat tietokoneet, joilla pääsevät scratchiin. Myös historian materiaalit tulee olla saatavilla, jotta sieltä voi tarkistaa faktoja.

Alkuun käydään pikainen kertaus historian ajanjaksosta, eli mitä kaikkea tuolla aikavälillä on tapahtunut ja mitä oppilailla on sieltä jäänyt parhaiten mieleen. Kerrataan myös edellisen ohjelmointikerran tuloksia, mitä tehtiin ja miten tehtiin, miten saatiin peli toimimaan. Katsotaan myös yhdessä erilaisia pelityyppejä, joita scratchilla voi ohjelmoida. Alussa jo opettaja muistuttaa, että yhteistyön tekeminen on erittäin suotavaa ja autetaan toinen toisiamme eteenpäin.

Oppilaat motivoituvat, koska saavat itse valita pelin aiheen ja suunnitella kuinka se toteutetaan. Toki joillekin voi tässä tulla hankaluuksia, mutta sitä varten opettajalla on valmiina joitakin aiheita, joita antaa oppilaille. Näin ei kellään mene koko aika pelin aiheen miettimiseen. Pelien ohjelmointi on myös lapsista mielenkiintoista ja palkitsevaa, joten senkin takia oppilaat varmasti mielellään tekevät tällaista tehtävää. Turhautumista ja hankaluuksia on varmasti luvassa, mutta tässäkin tulee korostaa yhteistyön merkitystä eteenpäin pääsemiseksi.

Tehtävä myös eriyttää oppilaiden taitotasojen mukaan, enemmän aiheeseen perehtyneet voivat ohjelmoida jo monimutkaisempia kokonaisuuksia, kun taas asiaa ehkä vielä arastelevat voivat keskittyä yksinkertaisempaan peliin.

Binääriluvut ja taikatemppuja, 6. lk

Krista Taipalvesi, CC BY-SA 4.0

8.10.2022

Krista Taipalvesi
LUMATIKKA syksy 2022

Binääriluvut ja taikatemppuja

Suunnittelin oppitunnit oman 6. luokkani käyttöön. Osa oppilaistani harjoittelee Tie koodariksi -sivustolla itsenäisesti ja osa on vasta sinut ScratchJr kanssa. Ajattelin, että binäärilukujen opettelu ja matemaattinen taikuus voisi olla hyvä yhteinen teema eivätkä taitoerot muodostuisi tehtävien tekemisen esteeksi. Oppitunnit suunnittelin pitäväni osana matematiikan oppitunteja.

Oppitunneilla harjoitellaan seuraavia Vihdin kunnan OPSiin kirjattuja 6. luokan oppimisen tavoitteita:

- Osaan esittää kysymyksiä ja tehdä perusteltuja päätelmiä havaintojeni pohjalta.
- Osaan käyttää ongelmanratkaisussa erilaisia strategioita.

Oppitunneilla oppilaat harjoittelevat siis mm. loogista päättelyä, säännön huomaamista ja säännön toimivuuden arviointia.

Oppimisen tavoitteiden arviointi on jatkuvaa eli opettaja havainnoi oppilaiden toimintaa oppitunnin aikana. Itsearviointi voidaan toteuttaa nopeasti vaikkapa peukku ylös – peukku alas arviointina oppitunnin päätteeksi.

1. **oppitunti: Tutustutaan binäärilukuihin.**
    - **a.** Selvitetään ensin oppilaiden ennakkotiedot: Mitä binääriluvut ovat? Missä niitä käytetään? (Lukujärjestelmä, jossa kaikki luvut muodostetaan käyttämällä vain lukuja 0 tai 1. Siksi luvuista voi tulla pitkiäkin. Esimerkiksi tietokoneet käsittelevät informaatiota binäärilukuina).
    - b. Harjoitellaan binäärilukuja ensin toiminnallisesti:
        - a. Harjoitellaan esittämään binäärilukuja ensin sormin ja sen jälkeen muodostetaan oppilasryhmistä arvuuteltavaksi binäärilukuja:
            1. Tähän idean löysin https://maol.fi/app/uploads/2019/10/Toiminnallisuus-matematiikan-opetuksessa-MAOL-ry-2019.pdf Sivuilta 4-5 löydät myös havainnollistavat kuvat opetuksesi tueksi.

2. Materiaalissa on myös vinkki binäärilukujen rakentamiseen linkkikuutioilla paikka-alustalle, mutta itse teetän Tuhattaituri 6b -kirjasta soveltuvat tehtävät sivulta 222-223. Oppikirjan tehtävissä mm. muodostetaan binäärilukuja värittämällä ja oppikirjassa on vielä pieni tietolaatikko binääriluvuista.

2. oppitunti: Taikuutta

Idean ja inspiraation löysin https://www.csunplugged.org/en/topics/error-detection-and-correction/parity-magic/ sivustolta.

a. Oppitunnille pitää ensin valmistaa 36 kaksipuoleista korttia. Korttien toinen puoli on musta ja toinen valkoinen. Saatavilla pitää olla myös yhtä monta taulumagneettia, jos kortit kiinnitetään taululle. (Printattavat kortit löytyvät yllä mainitulta sivustolta tai linkistä https://www.csunplugged.org/en/resources/parity-cards/) .
b. Oppilaille esitellään kortit ja kerrotaan, että nämä kaksipuoleiset kortit voisivat kuvata myös edellisellä oppitunnilla käsitellyn binääri eli 2-järjestelmän ominaisuuksia. Eli onko paikka-alustassa arvo valittuna vai ei tai että toinen väri kuvaa lukua 0 ja toinen lukua 1.
c. Seuraavaksi opettaja kertoo oppilaille oppineensa uuden taikatempun. Taikatemppua varten hän pyytää yhtä oppilaista tekemään 5x5 ruudukon taululle kortteja apuna käyttäen.
d. Tämän jälkeen opettaja kertoo vaikeuttavansa tulevaa taikatemppua ja lisää kortteja niin, että lopulta muodostuu 6x6 ruudukko.
e. Nyt opettaja pyytää yhtä oppilasta kääntämään jonkun kortin toisin päin. Opettaja ei katso, kun kortti käännetään. Osa oppilaista voi vielä varmistaa ettei opettaja taatusti huijaa ja osa painaa mieleensä mikä kortti käännettiin toisin päin.
f. Kaikkien yllätykseksi opettaja löytää käännetyn kortin ja kääntää sen takaisin alkuperäiseen asentoon. Tämä sama hyvä tuuri toistuu uudelleen ja uudelleen.
g. Tässä kohtaa pysähdytään ja keskustellaan oppilaiden kanssa siitä, että onko kyseessä vain satumaisen hyvä tuuri vai olisiko takana jokin sääntö tms.

h. Taikatempun sääntö on siis se, että kun opettaja teki 5x5 ruudukosta 6x6 ruudukon, hän lisäsi omat korttinsa niin, että jokaiselle pysty- ja vaakariville jäi parillinen määrä mustia ja valkoisia kortteja, jolloin oppilaan kääntämä kortti oli helppo huomata ”virheenä”.
i. Säännön löydyttyä oppilaita voi vielä pyytää esimerkiksi miettimään, että toimiiko taikatemppu, jos samaan aikaan käännetäänkin kaksi korttia? Kyllä/ei? Miksi näin on? Toimiiko temppu 7x7 ruudukossa tai vaikkapa 20 x 20 ruudukossa?
j. Tunnin voi halutessaan lopettaa hieman erilaiseen matemaattiseen taikatemppuun katsomalla taikuri Martti Sirénin videoita Tekniikan akateemiset TEKin Youtube -kanavalta esim. https://youtu.be/5G_RMOQtPkE

Mahdollisella kolmannella oppitunnilla oppilaat voisivat vielä pienissä ryhmissä pelata itsekin vastaavaa virheiden etsintäpeliä. https://www.csunplugged.org/en/topics/error-detection-and-correction/integrations/quick-card-flip-magic/

Videoita, pelejä ja Scratch, 6.-7.lk

Päivi Viinikka, CC BY-SA 4.0

1.5.2019

Tämä sopisi aloitukseksi esimerkiksi matematiikan tunnille oppilaille, joilla ei ole paljoa aiempaa ohjelmointikokemusta (lähinnä 6. tai 7. luokkalaiset). Tarkoituksena on harjoitella algoritmista ajattelua ensin pelien kautta ja sitten tutustua graafiseen ohjelmointiympäristöön.

Ensimmäisellä kerralla katsotaan video ja pelataan:

- Mitä se ohjelmointi oikein onkaan? Mietitään myös konkreettisia esimerkkejä missä ohjelmointia oikein hyödynnetään.
  https://youtu.be/eyD6V9–44E

- Pelataan ohjelmointilogiikkaa kehittäviä pelejä:
  - https://studio.code.org/hoc/1
  - http://lightbot.com/flash.html
  - https://codecombat.com/play

  Pelien kautta on tarkoitus oppia loogista ajattelua.

Toisella tunnilla lähdetään sitten ohjelmoimaan, kieleksi valitaan Scratch. Linkin (https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn=) sivustoa hyödyntäen voi valita oman projektin tai teemme yhdessä matopelin Scratchilla. Samalla, kun tutustumme graafiseen ohjelmointi ympäristöön keertaamme peruslaskutoimitukset, lukukäsitteen ja koordinaatistoa. Myös muuttuja käsite tulee tutuksi. Kotona vanhemmat/sisarukset/kaverit voivat sitten testata oppilaan tekemää peliä ja antaa palautetta tehdystä. Palautteen perusteella voidaan vielä miettiä lisäominaisuuksien lisäämistä peliin.

Scratch-projekti tutun tarinan pohjalta, alakoulu

Maria Jussila, CC BY-SA 4.0

6.1.2020

SCRATCH-projekti tutun tarinan pohjalta

Tavoitteena on ideoida pieni vuorokeskustelu parin kanssa, joka saadaan elämään Scratch-ohjelmointiympäristössä. Tehtävässä harjoitellaan parityöskentelyä ja Scratcissä tehdään aluksi ohjeen mukaan komentoja, jonka jälkeen tarkoituksena on että oppilaat soveltavat oppimaansa ja tekevät samantapaisia komentoja erilaisilla variaatioilla oman tarinansa pohjalta. Tarkoituksena on tehdä ohjelmointia tutuksi.

Ennakkotaidot:

Oppilaat ovat aiemmin toteuttaneet jonkun ohjeistettuun Scratch-projektiin ja osaavat seurata ohjelman antamia ohjeita. Oppilaiden pitäisi tietää perusperiaatteet siitä, miten komentoja asetellaan ja miten niitä etsitään. Oppilailla on oltava tunnukset, jotta projektit saadaan talteen.

Opettaja tarvitsee perustiedot Scratchin toiminnasta ja jos ohjelma ei ole kovin tuttu, opettajan kannattaa tehdä ensin itse tehtävä, niin tietää millaisiin haasteisiin oppilaat saattavat joutua.

Välineet/tarvikkeet:

Oppilaille tarvitaan paria kohden yksi tietokone.

Työskentelyn toteutus:

Oppilaat tutustuvat ensimmäisellä tunnilla Scratchin ohjeen avulla tarinan tekemistä. Kirjaudutaan Scratchiin ja aloitetaan uusi projekti. TUTORILAS-välilehdeltä valitaan ohje: CREATE A STORY ja tehdään pareittain sen ohjeiden avulla harjoitus. Näin kaikille on selvää perusperiaatteet siitä, mitä varsinaisessa tehtävässä on tarkoitus tehdä ja sen jälkeen voi suunnitella varsinaista toteutusta.

Kun oppilaille on selvinnyt, millaisia mahdollisuuksia tarinan kerrontaan on, opettaja voi antaa oppilaille tehtäväksi suunnitella jonkinlaisen vuorokeskustelun. Ideoinnin pohjana voi olla esim. joku reaaliaineessa käsitelty asia, joka pitää selittää keskustelun avulla. Tai keskustelu voi liittyä johonkin vuorovaikutustilanteeseen, joka pitää ratkaista. Keskustelu voi olla suoraan myös esimerkiksi jostain juuri käydystä tarinasta/sadusta tai siitä pitää tehdä variaatio. Keskustelu voisi olla myös esimerkiksi mainos, jos äidinkielessä on juuri käsitelty mainoksia. Tätä ideointia voisi tehdä ennen kuin seuraavan kerran asetutaan koneiden äärelle.

Seuraavalla koodauskerralla oppilaat kirjautuvat taas jommankumman parin tunnuksilla Scratchiin. Opettajan ohjeistuksen mukaan oppilaat luovat oman version. Oppilaat soveltavat aiemmin oppimaansa ja voivat palauttaa mieliinsä aiemmin ohjeen avulla tehdyn ohjelmoinnin avulla ydinkohdat tai soveltaa taas suoraan valmista mallia. Taitavammat voivat soveltaa oman osaamisensa mukaan lisää toiminnallisuutta.

Arviointi kun tehtävät ovat valmiita:

Jokainen pari kommentoi ”ilmoitukset ja kiitokset”- kohtaan muutamalla sanalla omaa työskentelyään. Tarinat julkaistaan ja arvotaan jokaiselle parille kolme paria, joiden tehtävää he käyvät katsomassa ja kommentoimassa. Opettaja käy myös kommentoimassa tarinoita. Ennen tehtävän aloitusta tehdään oppilaille selväksi, mitä arvioidaan. Opettaja tekee päätöksen arvioidaanko pelkästään sisältöä vai pelkästään teknistä toteutusta vai mahdollisesti molempia. Tähän opettajan pitää miettiä hyvät apukysymykset, jotta oppilaat tietävät, mitä arvioidaan.

Idean saa jakaa

Tuntisuunnitelmat ohjelmointitunneille Scratchillä, alakoulu

Outi Liukkonen, CC BY-SA 4.0

6.1.2020

Tuntisuunnitelmat ohjelmointitunneille Scratchilla

Tunti 1

- Pieni avaus miksi ohjelmointia ja katsotaan työskentelyyn liittyvät tavoitteet ja jakson rakenne
- Tutustutaan Scratchilla tehtyihin videoihin ja peleihin open kasaamien linkkien avulla (ja edistyneemmät voi laajentaa tutkimuksiaan valmiiksi valittujen linkkien ulkopuolelle)
- Kurkataan milta Scratch-editori näyttää ja miten hahmon saa liikkumaan, miten hahmoa ja taustaa vaihdetaan (söpöys motivoi osaa oppilaista)
- Luodaan omat tunnukset

Tunti 2

- Kirjaudutaan omaan Scratchiin ja tehdään ohjatusti ensimmäinen säie ”kun klikataan” ja joku lause
- Kokeillaan hetki komentoja itsenäisesti
- Tehdään ohjatusti silmukka
- Kokeillaan hetki omaa silmukkaa
- Katsotaan valmiiden ohjelmien sisälle

Tunti 3

- Käydään läpi oman ohjelman tavoitteet
- Tehdään ohjatusti ehtolause
- Kokeillaan oman ehtolauseen rakentamista
- Omaa ohjelmointia

Tunti 4

- Muuttujat yhteisesti
- Kokeilua
- Muita esiin nousseita asioita yhteisesti
- Omaa ohjelmointia

Tunti 5

- Omaa ohjelmointia

Jakson tuotos					
5	6	7	8	9	10
Säie aloitettu ”kun klikataan” -palikalla		Silmukka	Silmukka, jonka jälkeen tulee joku muu komento	Kaksi säiettä	Säikeitä kolme tai enemmän
Lause	Kaksi eri lausetta	Kolme lausetta	Ääni tai ulkonäkö -lausetta käytetty		
Lause, joka ei ole suoraan valikosta poimittu	Kaksi eri lausetta, jotka eivät ole suoraan valikosta poimittu	Kolme lausetta, jotka eivät ole suoraan valikosta			
Taustakuva	hahmo, joka ei ole scratsin tunnus kissa		Kaksi hahmoa		
		Ehtolause	Ehtolause, jonka sisällä useampi lause	Useampi ehtolause	Useampi ehtolause, jossa käytetty sinisiä ja vihreitä ”timantteja”
				Muuttuja jolla on järkevä tarkoitus	
				Joku itse keksitty toiminto	Useampi itse keksitty toiminto
Työskentely					
5	6	7	8	9	10
Tekee palkollisen, mutta on mielummin esimerkiksi puhelimella eikä halua haastaa itseään.	Tekee muutaman jutun ihan neutraalisti, mutta ei sitten intoudu yrittämään yhtään enempää.	Työskentelee jokaisella tunnilla jonkun verran, mutta ei yritä ylittää itseään.	Käyttää kaikki oppitunnit asian opetteluun ja yrittää ihan tosissaan.	Työskentelee aktiivisesti projektin parissa. Kokee oivalluksia ja ideoi omia juttuja.	Ihan intopiukeena omallakin ajalla ideoi tai koodaa ohjelmaa.

Ohjelmoinnin alkeet -kokonaisuus, alakoulu

7.4.2021

Ohjelmoinnin alkeita

Nämä tunnit on tarkoitettu oppilaille, jotka eivät ole vielä ohjelmoineet mitään, mutta jotka osaavat lukea ja kirjoittaa, ja joille tietokoneet ovat vähän tuttuja. Tunneilla nostetaan esille myös luonteenvahvuuksia, ja oletuksena on, että tällainen vahvuusperustainen opetus on oppilaille jo ennalta tuttua. Vahvuusopetuksen voi opettaja sisällyttää tunteihin oppilaiden aiemman opitun mukaan.

Näiden kahden oppitunnin tavoitteena on ohjelmoinnin perusteiden ymmärtäminen ja yhdessä työskentely. Tunneilla opitaan mm. sinnikkyyttä, luovuutta, toisten huomioon ottamista, ryhmätyötaitoja, rehellisyyttä, keskittymistä ja ohjeiden noudattamista. Tavoitteena on myös, että ohjelmointi koetaan helppona ja hauskana, ja että kaikilla syntyy halu lähteä rohkeasti kokeilemaan ja harjoittelemaan ohjelmointia jatkossa. Tunnit eivät varsinaisesti ole puhtaasti minkään aineen oppitunteja. Niissä yhdistyy on liikunta, äidinkieli, matematiikka ja vahvuusopetus. Tunteja ei ole tarkoitus pitää peräkkäin tai samana päivänä, jottei saman asian toistoa tule liikaa.

Ensimmäinen oppitunti

Tarvikkeet: kynä ja paperia jokaiselle ryhmälle

Robotti-leikki, opettaja robottina: Opettaja esittää robottia, joka tottelee oppilaiden sille antamia käskyjä yksi käsky kerrallaan.

Ensin opettaja kertoo ohjeet: Oppilaat antavat robotille käskyjä vuorotellen siten, että robotti onnistuu suorittamaan oppilaiden yhdessä sopiman tehtävän (ikkunan avaaminen, nelikulmion piirtäminen taululle jne.). Hän ohjeistaa, että robotti ymmärtää vain yhden ohjeen kerrallaan (kävele kaksi askelta eteenpäin / nosta oikea käsi / tartu kiinni kahvasta/ käännä oikeaan jne.). Jos ohje on vääränlainen, operobotti ei liiku. Opettaja menee tehtävän sopimisen ajaksi ulos, jotta ei kuule sitä. Näin tehtävästä tulee oppilaille jännempi.

Arviointi: Leikin jälkeen käydään läpi robottien ohjelmointikieli: Millaiset käskyt eivät toimineet? Millaiset käskyt toimivat? Millaisia käskyjen pitää olla, että päästään oikeaan lopputulokseen?

Robotti-leikki: Oppilaat robottina

Oppilaat muodostavat kolmen hengen ryhmät. Vuoron perään yksi ohjaa robottia tekemään jonkun tehtävän, toinen on robotti ja kolmas ilmoittaa virheestä (esim. jos käsky on liian monimutkainen tai robotti tekee muuta kuin on käsketty). Kukin ryhmä valitsee lopuksi hauskimman/onnistuneimman ohjelmointinsa ja kirjaa kyseilelle robotille annetut ohjeet ylös. Voidaan toteuttaa myös paritehtävänä, jolloin tarkkailija jää pois.

Arviointi: Jokainen ryhmä esittelee kirjoittamansa ohjeet muille. Käydään yhdessä kirjoitetut ohjeet läpi: Ymmärtävätkö muut ohjeista, mitä robotin tehtävänä alunperin oli? Ovatko ohjeet onnistuneet? Löytävätkö muut ohjeista puutteita?

Miten yhteistyö sujui? Jaksoivatko kaikki loppuun saakka? Toimivatko kaikki ohjeen mukaan? Miltä leikki tuntui? Mitä vahvuuksia onnistumisessa tarvittiin?

Toinen oppitunti

Tarvikkeet: maalarinteippi, ipadit tai vastaavat kaikille. Opettajan on hyvä tutustua linkkien peleihin etukäteen.

Robotit ruudukossa

Siirretään robotit kulkemaan ruudukossa. Käytävän tai liikuntasalin lattiaan on tehty maalarinteipillä ruudukoita (sopivan kokoisia, sopiva määrä, riippuen luokan koosta ja oppilaiden iästä). Ruudukossa on erilaisia esineitä. Robotti kertoo, minkä esineen haluaa noutaa, ja pari ohjelmoi robotin sinne käyttämällä käskyjä “mene yksi eteenpäin, käänny oikealle” jne. Paperilla voi myös olla valmiita käskysarjoja ja pari yrittää ensin ohjeet lukemalla päätellä, minkä esineen luokse käskyt ohjaavat. Sen jälkeen robotti toimii ohjeen mukaan ja huomataan, oliko arvioitu kohde oikea.

Tehtävästä voi myös varioida kilpailun: Kuka pääsee ensin kohteensa luo tai kohteelle ja takaisin tuoden esineen kotipesään. Jos kaksi robottia osuvat vierekkäisille ruuduille, molemmat joutuvat aloittamaan alusta. Sääntöjä voi kehittää luokan tason mukaan.

Arviointi: Osattiinko antaa niin hyviä ohjeita, että robotit menivät optimaalista reittiä perille? Yrittikö robotti ohittaa vääriä ohjeita tavoitteen saavuttamiseksi? Miten yhteistyö parin kanssa sujui? Mitä vahvuuksia tässä tehtävässä tarvittiin? Osattiinko niitä käyttää? Oliko joku erityinen asia, jonka takia parin tehtävä onnistui?

Tietokone mukaan ohjelmointiin

Pelataan ohjelmointilogiikkaa kehittäviä pelejä:

–https://studio.code.org/hoc/1.

–https://codecombat.com/play

Ensimmäisessä linkissä on aluksi mukava lyhyt ohjelmointiin motivoiva video, joka kannattaa katsoa yhdessä luokan kanssa, jos tässä vaiheessa on siirrytty luokkaan (voi olla hauska myös jäädä saliin ja pelata koneilla vaihteeksi siellä).
Sen jälkeen pelataan jompaa kumpaa peliä taitojen mukaan (ensimmäinen peli on helpompi). Pelien kautta on tarkoitus oppia loogista ajattelua tulevia ohjelmointeja ajatellen.

Arviointi: Onnistuivatko oppilaat ymmärtämään tietokonepelin idean, kun olivat harjoitelleet samaa asiaa ensin roboteilla? Oliko peli-idean ymmärtäminen joillekin vaikeaa? Voiko tästä jatkaa itse ohjelmointiin?

Lähteenä tuntisuunnitelman tehtävissä Innokas-materiaali.

Ruotsin alkeita Scratchissa, alakoulu

3.10.2021

Aihepiirin valinta ja rajaus

Yhdistän tässä tuntisuunnitelmassa ruotsin kielen alkeita visuaaliseen ohjelmointiin. Tavoitteena on tutustua Scratch-ohjelmointiympäristöön, rakentaa animaatio vähintään kahden hahmon kohtaamisesta ja kerrata samalla juuri opiskeltuja ruotsinkielisiä tervehdyksiä ja fraaseja. Havaintoni mukaan ne oppilaat, jotka eivät ole erityisen motivoituneita ruotsin opiskelijoita, ovat sitäkin innokkaampia tietokoneiden kanssa.

Oppimistavoitteet

1. ruotsi: oppilas muistaa perustervehdykset ja fraasit, sekä osaa käyttää ja yhdistellä niitä järkevästi

2. ohjelmointi: oppilas saa Scratch-hahmon liikkumaan ja puhumaan, käyttää kosketus-ehtoa ja odota-komentoa

3. yhteistyötaidot: oppilas osaa toimia aktiivisesti yhteistyössä parin kanssa

4. oppilas osaa antaa rakentavaa palautetta

Arviointi toteutetaan vertaispalautteena, jolloin pari "koeajaa" ohjelman, tarkistaa ruotsinkielisten sanojen ja fraasien oikeellisuuden ja antaa sanallisen palautteen.

Scratch-ympäristössä oppilas näkee helposti, kun joku ei toimi. Silloin neuvotaan käymään komennot yksi kerrallaan läpi ja kokeilemaan rohkeasti uudelleen. Oppilaat saavat myös auttaa toisiaan.

Työskentelyvälineet ja opetusmenetelmät

Työskentelyyn tarvitaan ruotsin oppikirja, tietokoneet, hiiret, (kuulokkeet) ja esitystekniikkaa.

Läksynä on ollut tunnusten luominen Scratchiin huoltajan luvalla. Opettaja näyttää uuden projektin aloittamisen, hahmon luomisen ja taustan vaihtamisen. Osaa oppilaista motivoi visuaalinen ympäristö, ja he saisivat kulutettua koko kaksoistunnin pelkästään tähän, mutta muutaman minuutin kuluttua opettaja näyttää malliohjelman, jonka sisään kurkistetaan nopeasti. Katsotaan yhdessä, mistä löytyy liikkumiskomennot ja miten hahmot saa puhumaan. Osa motivoituu tietokoneella työskentelystä ja näppärille voi antaa luvan tehdä animaatioon jonkun yllätyksen tai jäynän.

Oppilailla on lupa neuvoa ja auttaa toinen toisiaan, mutta kuitenkin niin, että jokainen tekee oman työn itse. Eli toiselle saa näyttää mitä palikkaa kannattaa käyttää ja mistä se löytyy, mutta tekijä itse syöttää tarvittavat arvot ja raahaa palikat paikalleen.

Taitavat oppilaat voivat maustaa tehtävää omilla ideoilla: laittaa hahmon liikkumaan valmiiksi määrätyn reitin sijaan hiirellä tai nuolinäppäimillä, laittaa hahmojen väliin esteitä, jotka pitää ohittaa, tms.

Tehtävän on tarkoitus toimia aloituksena visuaalisen ohjelmoinnin kokonaisuudelle. Opittuja komentoja (aloitus, liikkuminen, sano, kosketus, odota) tarvitaan myös seuraavissa projekteissa.

Algoritmisen ajattelun harjoittelua eri oppiaineissa, alakoulu

3.3.2022

Algoritminen ajattelu, alakoulu

Aihepiirin valinta ja rajaus

Suunnittelemani ohjelmoinnin oppimiskokonaisuuden ensimmäiset oppitunnit voidaan käyttää matematiikan, äidinkielen ja ympäristöopin oppiaineissa. Kolmannella tunnilla tehdään kuvataidetehtävä ja neljännen tunnin voi tehdä myös liikuntatunnilla. Suunnitelmaa tehdessäni minulla ei ollut omaa luokkaa opintovapaani vuoksi ja mietin kokonaisuutta, jota voisin opettaa varhaisteini-ikäisille lapsilleni ja heidän luokilleen (luokat 6-7). En tiedä, millaista opetusta he ovat aikaisemmin saaneet, joten halusin aloittaa pohjustamalla aihetta helpoin tehtävin. Ajatuksena on saada oppilaat miettimään, miten heistä kerätään tietoja, millainen digitaalinen jalanjälki heistä muodostuu ja miten yksinkertainen lajittelu- ja suosittelualgoritmi toimii.

Ensimmäinen oppitunti

Tavoitteet

- Tavoitteena on selvittää, mitä oppilaat jo tietävät aiheesta ja kerrata, mitä algoritminen ajattelu on.
- Tavoitteena on osallistua keskusteluun yhteisistä asioista.
- Tavoitteena on harjoitella yksinkertaisten käskyjen antoa.
- Tavoitteena on harjoitella ohjaamaan toista oppilasta.
- Tavoitteena on kuunnella ja keskittyä ohjeisiin.
- Tavoitteena on harjoitella toimimista yhteistyössä muiden oppilaiden kanssa.

Työskentelyvälineet

- tila, jossa voidaan yhdessä katsoa virittelyvideo
- teippi, jolla aikuinen merkitsee lattiaan ruudukot
- erilaisia ruudukkoihin laitettavia esineitä

Oppitunnin rakenne

- Katsotaan yhdessä **virittelyvideo** https://www.youtube.com/watch?v=j-6N3bLgYyQ. Keskustellaan oppilaille videosta heränneistä ajatuksista.
- Opettaja kertoo, mitä tunnilla harjoitellaan ja mikä on tavoitteena.
- **Harjoitellaan selkeiden ohjeiden antamista opebotille**. Opettaja on kirjoittanut taululle tehtäviä ja oppilaat valitsevat niistä, mitä opebotin tulee tehdä (esim. Kävele kaapille/tiskialtaalle/dokumenttikameran luo. Avaa kaappi. Laita kirja hyllyyn.) Opettaja seisoo luokan ovella ja odottaa oppilaiden ohjeita. Oppilaiden tulee antaa riittävän lyhyitä ja selkeitä ohjeita, jotta

opebotti suorittaa tehtävän. Liian monimutkaisen ohjeen annon jälkeen opebotti sanoo “en ymmärrä”.
- Lattiaan on teipattu monta 5x5 ruudukkoa. Oppilaan pitäisi mahtua seisomaan ruudussa. Ruudukkoon laitetaan joitain esineitä. Ruudukkojen lukumäärästä riippuen toimitaan pareittain tai ryhmässä. Yksi oppilas seisoo jossain aloitusruudussa ja **toinen oppilas ohjaa hänet tietyn esineen luo käyttäen käskyjä eteenpäin, taaksepäin, käänny oikealle ja käänny vasemmalle**.
- Keskustellaan tunnista lopuksi.

Toinen oppitunti

Tavoitteet

- Tavoitteena on harjoitella algoritmin kirjoittamista.
- Tavoitteena on oppia, mikä on suosittelualgoritmi.
- Tavoitteena on miettiä, kuinka paljon tietoa oppilaasta tallentuu erilaisiin palveluihin.
- Tavoitteena on harjoitella suosittelualgoritmin kirjoittamista.
- Tavoitteena on kuunnella ja keskittyä ohjeisiin.
- Tavoitteena on harjoitella toimimista yhteistyössä muiden oppilaiden kanssa.

Työskentelyvälineet

- paperi tai vihko muistiinpanoja varten
- tila, jossa voidaan yhdessä katsoa opetusvideo

Oppitunnin rakenne

- Kerrataan, mikä algoritmi on.
- Opettaja kertoo, mitä tunnilla harjoitellaan ja mikä on tavoitteena.
- **Harjoitellaan kirjoittamaan yksinkertaisia ohjeita ja tehdään yhdessä yksi ohje taululle.**
- **Parit tai ryhmät kirjoittavat vihkoihinsa algoritmin opettajan valitsemiin tehtäviin** (esimerkiksi repun avaaminen, penaalin ottaminen repusta, kynän ottaminen penaalista, kirjan laittaminen repusta laatikkoon, repun sulkeminen).
- **Oppilaat vaihtavat vihkojaan toisen oppilaan kanssa ja toimitaan erilaisen ohjeen mukaan.**
- **Katsotaan video suosittelualgoritmeista** https://youtu.be/3wvdKsVjEK0. Jos koko videon katseluun ei ole aikaa (12:22 min), voisi siitä katsoa kohdan 1:35-10:29.
- Keskustellaan videon herättämistä ajatuksista ja tietojen keräämisestä.
- **Mietitään yhdessä, millainen voisi olla oppilaille tutun palvelun suosittelualgoritmi.** Esimerkiksi Spotify. Vaikuttaako käyttäjän sukupuoli?

Käyttäjän ikä? Asuinmaa tai asuinpaikka? Käyttäjän aikaisemmat valinnat? Käyttäjän ystävien valinnat?
- **Parit tai ryhmät kirjoittavat vihkoihinsa, millainen voisi olla suosittelualgoritmi yhdessä heidän käyttämässään palvelussa.** (Tiktok, Netflix, verkkokaupat jne.)

Kolmas oppitunti

Tavoitteet

- Tavoitteena on miettiä, kuinka paljon tietoa oppilaasta tallentuu erilaisiin palveluihin.
- Tavoitteena on harjoitella mallista piirtämistä.
- Tavoitteena on kuunnella ja keskittyä ohjeisiin.

Työskentelyvälineet

- paperi kuvataidetehtävää varten (myös vihko käy)

Oppitunnin rakenne

- Kerrataan, mitä edellisellä tunnilla puhuttiin tietojen keräämisestä.
- Opettaja kertoo, mitä tunnilla harjoitellaan ja mikä on tavoitteena.
- **Piirretään paperille/vihkoon oman jalan ääriviivat.** Mietitään, millainen digitaalinen jalanjälki oppilaasta syntyy eli mihin kaikkiin sovelluksiin ja palveluihin oppilaasta kerätään tietoa.
- **Oppilas piirtää jalanjälkeensä kaikkien niiden sovellusten ja palveluiden logot/tunnukset, joita hän käyttää.**

Neljäs oppitunti

Tavoitteet

- Tavoitteena on kerrata prosentti-, desimaali- ja murtolukuja.
- Tavoitteena on harjoitella matemaattisen ajattelun monipuolista käyttöä.
- Tavoitteena on oppia soveltamaan prosenttilaskun käsitettä ja harjoitella ymmärtämään yhteys aiemmin opittuun.
- Tavoitteena on oppia, miten lajittelualgoritmi toimii.
- Tavoitteena on kuunnella ohjeita ja toimia niiden mukaan.
- Tavoitteena on harjoitella toimimista yhteistyössä muiden oppilaiden kanssa.

Työskentelyvälineet

- esimerkiksi paperille/käytetyille lahjakassipaloille/pahville merkittyjä prosentti-, desimaali- ja murtolukuja (kaikki eriarvoisia), esimerkkejä liitteessä 1
- liitu, teippi tai urheiluvaraston välineitä lajittelujärjestelmäkaavion merkintään
- Liitteessä 2 on lajittelujärjestelmäkaavio 6, 8 ja 10 oppilaan ryhmään.

Oppitunnin rakenne

- Opettaja muistuttaa mieliin, että edellisellä kerralla oppilaat tutustuivat suosittelualgoritmiin. Tällä kertaa he saavat tutustua yksinkertaiseen lajittelualgoritmiin ja toimia itse koneistona.
- Opettaja kertoo oppilaille tunnin tavoitteet ja rakenteen.
- Opettaja jakaa oppilaat kahteen ryhmään oppilasmäärästä riippuen.
- **Toinen ryhmistä voi järjestää liitteen 1 mukaisia lukuja suuruusjärjestykseen eri tavoin tai leikkiä arvaa lukuni leikkiä** (jossa yritetään saada selville toisen luku esittämällä erilaisia kysymyksiä)**.**
- Opettaja ohjaa lajittelualgoritmiryhmää. **Hän antaa jokaiselle ryhmän jäsenelle prosentti-, desimaali- tai murtolukupaperin ja osoittaa oppilaalle hänen aloituspaikkansa lajittelukaavion alussa.** Oppilaiden aloitusjärjestyksellä ei ole väliä.
- Opettajan antaessa merkin kaikki oppilaat **siirtyvät seuraavaan** ruutuun/ympyrään, jossa kaksi oppilasta kohtaavat.
- Oppilaat tervehtivät toisiaan ja **vertaavat yhdessä käsissään olevia lukuja.** Se oppilaista, jolla on pienempi luku, seuraa nuolta/reittiä vasemmalle ja toinen oppilas, jolla on suurempi luku, seuraa nuolta/reittiä oikealle seuraavaan ruutuun/ympyrään.
- Oppilaat **päätyvät jälleen uuteen ruutuun**, jossa on uusi oppilas, jonka kanssa omaa lukua verrataan uudelleen.
- Näin kauan jatketaan, kunnes nuolta seuraamalla kaikki päätyvät lopetusruutuihin/-ympyröihin.
- **Opettaja pyytää oppilaita näyttämään prosentti-, desimaali- ja murtolukunsa. Tarkistetaan, että kaikki ovat oikeilla paikoillaan.**
- Näin toimii yksinkertainen lajittelualgoritmi.
- **Oppilasryhmät vaihtavat osia ja opettaja saa uuden lajittelualgoritmiryhmän.**
- Lopuksi keskustellaan lukujen vertailusta omatoimisessa ja opettajan vetämässä ryhmässä.

Arviointi

Opettaja havainnoi oppilaiden toimintaa oppituntien aikana. Tuntien lopuksi keskustellaan yhdessä. Koulussamme on käytössä Qridi, johon oppilaille voisi tehdä itsearvioinnin tai tavoitteiden saavuttamista voidaan arvioida peukku ylös tai alas näyttämisellä.

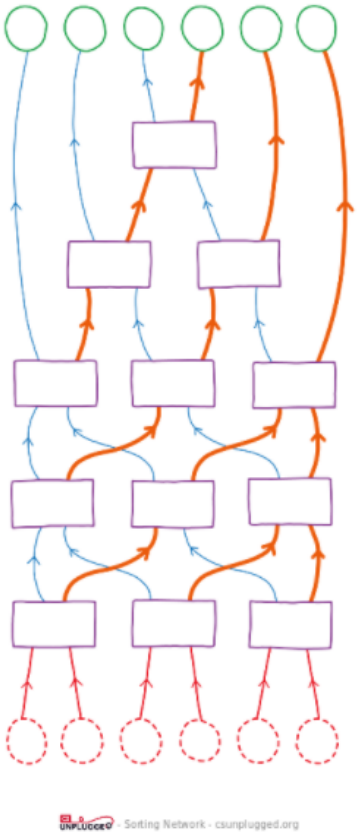
Lähteet

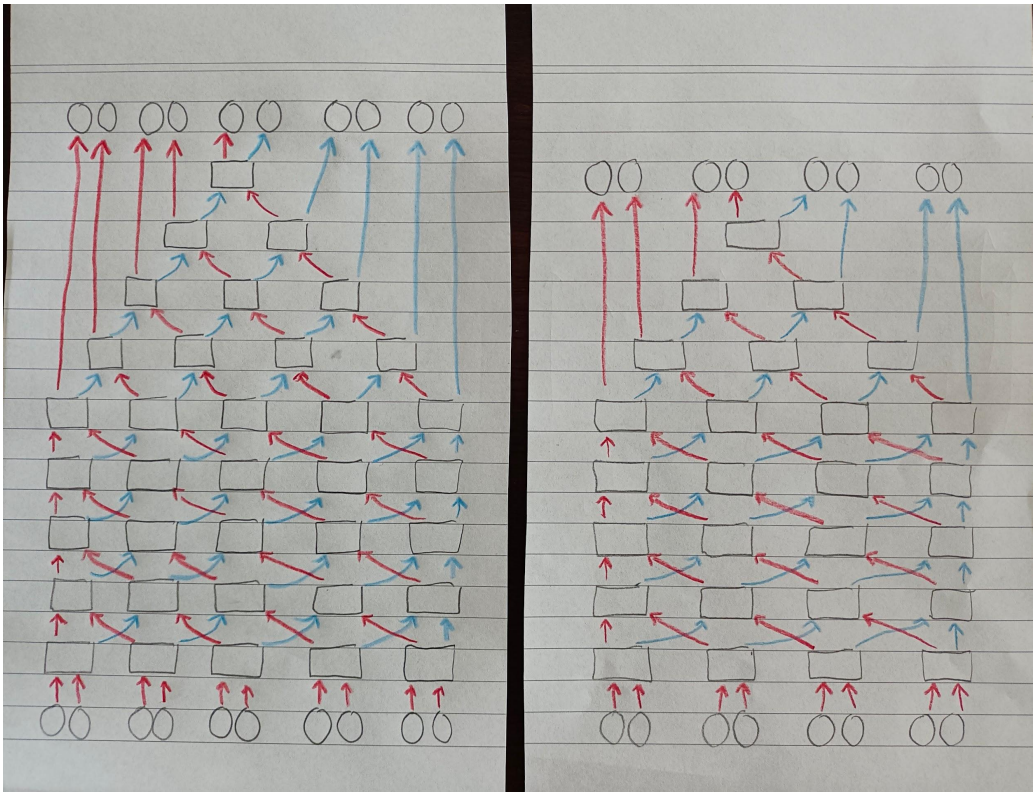
- https://www.cs.helsinki.fi/group/linkki/materiaali/lumatikka-ladattavat/.piilotettu kansio/tuntisuunnitelmapankki.pdf (Luettu 18.2.2022)
- https://www.youtube.com/watch?v=j-6N3bLgYyQ (katsottu 18.2.2022)
- https://www.innokas.fi/wp-content/uploads/2019/02/Ohjelmoinnillinen_ajattelu_-teht%C3%A4v%C3%A4kortti.pdf (Luettu 18.2.2022)
- https://mooc.helsinki.fi/pluginfile.php/206091/mod_folder/content/0/ohjelmointi-ja-internet-harjoituksia.pdf (Luettu 18.2.2022)
- https://www.csunplugged.org/en/topics/sorting-networks/unit-plan/reinforcing-numeracy-through-a-sorting-network-junior/ (Luettu 16.2.2022)
- https://www.csunplugged.org/en/resources/sorting-network/ (Luettu 16.2.2022)

Liite 1.
Esimerkkejä murto-, desimaali- ja prosenttilaskutehtäviin:

1/100	1/10	1/3	1/2	3/4	20/100	4/10
13/50	58/100	8/10	86/100	91/100	1/4	13/50
96/100	0,05	0,09	0,13	0,26	0,35	0,44
0,52	0,60	0,69	0,77	0,81	0,89	0,93
0,97	1,01	1,1	7%	18 %	27 %	39 %
45 %	49 %	56 %	62 %	70 %	79 %	83 %
90 %	94 %	100 %	120 %			

Liite 2





Halloween-teemaista ohjelmointia BeeBot-roboteilla, alakoulu

Nimetön,

19.10.2022

Tuntisuunnitelma
Algoritmisen ajattelun kehittäminen
Vaativan erityisen tuen pienryhmä

Tunti toteutetaan matematiikan tunnilla ja tavoitteena on tutustua ohjelmoinnin perusteisiin ja kasvattaa kiinnostusta aihetta kohtaan. Motivointi tehdään tulevan juhlapyhän juhlimisen varjolla. Oppimistavoitteena on, että kiinnostus syntyy aihetta kohtaan sekä ohjelmoinnin perusteiden ymmärtäminen. Työskentelyä arvioidaan kokonaisuutena laaja-alaisen osaamisen kannalta sekä matematiikan oppiaineen kannalta ohjelmoinnin osa-alueetta. Oppilas arvoi lopussa omaa osaamistaan sanallisesti kertomalla, miten tehtävä sujui. Työskentelyssä kaikki oppilaat tekevät yhteistyötä (jos olisi isompi ryhmä, tehtävä suoritettaisiin esimerkiksi 3 oppilaan ryhmissä) ja opettajan tehtävänä on rohkaista kaikkia puhaltamaan yhteen hiileen.

Kesto: 45 minuuttia
Ryhmän koko: 3 oppilasta, joten yksi beebot on tarpeeksi. Isommalle ryhmälle tarvitsi enemmän robotteja tai tämä voi toimia vaikka kiertopisteen pisteenä.

Oppilaani ovat isompia, mutta erityisiä, joten ohjeiden anto on mietitty heidän osaamisensa mukaan. Voi olla että niitä pitää muokata oppilasryhmän mukaan.

Tunnin alku:
Tunnin alussa esitellään luokkamme uusi jäsen, Beeba Beebot. Kerrotaan, että Beeba on ampiaiseksi Halloweenin kunniaksi pukeutunut robootti. Beeba on tullut pitkän matkan, aina Robostaniasta saakka. Kerrotaan, miten Beeba toimii (mistä napista tapahtuu mitäkin). Annetaan tehtävänanto (voi laittaa näkyville taululle).

Tarvikkeet:
Ruudukko Beebotia varten (Tausta voi olla valkoinen tai vaikka ihan vaan lattia, tai sitten ruudukon taustan voi tehdä vaikka tieksi)
Beebot
Laminoidut Halloween-kortit (liitteenä kortit tulostettavaksi)

Tehtävänanto:
Beeban Halloween koristeet ovat tippuneet pitkin poikin pihaa.

1. Ilmoita ensin aikuiselle, mitä koristetta lähdet tavoittelemaan.

2. Suunnittele paperille, mitä nappeja aiot painaa, jotta pääset tavoittelemaasi ruutuun.
3. Paina suunnittelemiasi nappeja ja auta Beeba keräämään tippuneet koristeet ruudukosta.

HOX! Jos ajelet jonkun koristeen yli, ei hätää, se jää koskemattomana odottamaan kunnes pääset juuri sen kohdalle.

Tunnin toiminta(keskivaihe):
Oppilaat ohjaavat beebottia ruudukon läpi. Päästyään tavoiteltuun ruutuun oppilas kuvan pois ruudukosta ja valitsee seuraavan kohteen.

Tunnin lopetus:
Käydään läpi saalis ja keskustellaan siitä, miten tehtävä sujui. Kerrotaan, että Beeba on nyt iloinen ostoksistaan, kun pääsee vihdoin koristelemaan kotipesänsä. Koristellaan luokka halloween-koristeilla.

Vaihtoehtoinen toteutus muuhun, kuin Halloweenin aikaan:
Jos Beeban pukeutuminen ampiaiseksi ei sovi sen hetkiseen vuodenaikaan, voi beebotin koristella ennen tehtävään siirtymistä yhteisesti papereiden yms. avulla. Tausta ja kerättävät asiathan voivat olla mitä vaan, kun vähän muokkaa tehtävän ohjeistusta.

Mielestäni tämä tunti sopisi esimerkiksi ohjelmointi-jakson aloitustunniksi, joten tulevilla tunneilla voisi keskittyä siirtämään opittua arkielämänkin tilanteisiin. Tunnin lopussa voisi yhdessä myös pilkkoa sitä, miten jokainen meistä jokainen tavallaan suorittaa toiminnoissaan tietynlaista ohjelmointia.

Tarinan kuvittaminen Scratchilla, alakoulu

10.11.2022

Tämä tehtävä vaatii sen, että Scratch on oppilaille jo jonkin verran tuttu.

Oppilaat kirjoittavat lyhyen tarinan äidinkielen tunnilla. Scratchillä koodataan tarina kuvina. Riittää, että pääasiat näkyvät.

Oppilaiden tavoitteena on oppia löytämään tekstistä keskeisimmät asiat. Toinen tavoite on miettiä, miten asiaa voisi visualisoida helposti, jotta koodaus onnistuu. Tavoitteita arvioidaan erillään. Oppilas voi esimerkiksi sanallisesti selittää hienosti, mitä haluaisi tehdä, vaikka koodaustaito ei vielä riitä.

Oppilaat pääsevät esittämään tarinansa ja visualisoinninsa muille oppilaille tai mahdollisesti kummioppilaille. Oppilaat saavat näin myös positiivista vertaispalautetta muilta ja saavat ideoita tulevaisuuden koodaustuokioihin. Päävastuu arvioinnista on kuitenkin opettajalla.

Opetuksessa käytetään tietokoneita. Oppilaat voivat kirjoittaa tarinan joko käsin tai koneella. Sen jälkeen he koodaavat Scratch ympäristössä. Koodaaminen on usein oppilaiden mielestä kivaa, joten osaa motivoi ihan vain se. Osaa voi motivoida mahdollisuus lukea tarina kummioppilaille. Oppilaita kannattaa kannustaa keskustelemaan koodauksen aikana, sillä usein kaverit keksivät koodauspulmiin ratkaisun. Ideoinnin pohjalle opettajan on hyvä näyttää esimerkki, sillä tekstistä tehtävä visualisointi voi olla oppilaille tuttua vain kuvissa. Scratchissa voi koodata hahmoja myös liikkumaan tai vaihtamaan ilmettä. Scratchin käyttötaidot auttavat tässä projektissa ja tämä tehtävä auttaa koodaamaan monipuolisempia tuotoksia tulevaisuudessa. Erityisesti hahmon asun vaihto, taustan vaihto ja äänien käyttö tulevat todennököisesti tutuksi ja näiden muuttamistä opettajan on hyvä demostroida oppilaille. Toki osaajat voivat suoraan siirtyä oman tehtävän pariin.

Micro:bit alkeet käsitöissä, 6.-8. lk

Tiina Torvinen, CC BY-SA 4.0

6.1.2020

24. MARRASKUUTA 2019

MICRO:BIT ALKEET

TUNTISUUNNITELMA

TIINA TORVINEN

Sisällysluettelo

Aihepiirin valinta ja rajaus 1
Kohderyhmä 1
Oppimistavoitteet 1
Eriyttäminen 1
Arviointi 1
Sisältö 2
Aikataulutus, opetusmenetelmät, materiaalit, oppimisympäristöt 2
Tuntitehtävät 3
Käsityöprojektin ideointitehtävä 3
Huomioita 3
Lähteet 3
Liitteet 4

Aihepiirin valinta ja rajaus

Opetuksen aiheena on ohjelmoinnin alkeet Micro:bit -mikrokontrollerilla sen visuaalisessa MakeCode -ohjelmointiympäristössä. Ympäristössämme on lukuisia esineitä, joissa on käytetty ohjelmointia. Ohjelmoinnin alkeiden opetuksessa lähdetään liikkeelle ympäristöstä löytyvistä tavanomaisista esineistä, joiden toiminta on tarkoitus toistaa koodina. Toisin sanoen tavoitteena on saada Micro:bitillä toistettua tietyn tavanomaisen esineen ohjelmoitu toiminne.

Ohjelmoinnin alkeiden opetuksen tarkoituksena on valmistautua käsityöprojektiin, jossa tehtävänä on valmistaa prototyyppi tuotteesta, jossa on hyödynnetty ohjelmointia ja Micro:bit -mikrokontrolleria.

Kohderyhmä

Tuntisuunnitelma on tarkoitettu ensisijaisesti 7. luokan monimateriaalisen käsityön oppitunnille (2 x 45 min). Sitä voi soveltaa myös 6. ja 8. luokan käsityössä tilanteesta ja ryhmän osaamistasosta riippuen.

Oppimistavoitteet

Oppilaat tietävät, millaisia komentoja on missäkin osiossa ja osaavat käyttää visuaalista ohjelmointiympäristöä koodin rakentamiseen sekä siirtää ohjelman Micro:bitille. Tavoitteena on, että Micro:bit ja sen mahdollisuudet tulevat oppilaille tutuiksi, jotta he pystyvät ideoimaan käsityöprojektia oppituntien päätyttyä ja aloittamaan projektin suunnittelun viimeistään seuraavalla oppitunnilla. Lisäksi pyritään vahvistamaan oppilaiden ryhmätyötaitoja.

Eriyttäminen

Oppilaille, joilla on aiempaa kokemusta ohjelmoinnista, on mahdollisuus kertoa kokemuksistaan ja osallistua vertaisten opetukseen yhteisen keskustelun, tuntitehtävien ja ryhmätyönä tehtävän käsityöprojektin kautta. Yhden tai useamman oppilaan aiempi ohjelmointiosaaminen mahdollistaa ryhmälle haasteellisempaan projektiin tarttumisen. Riippuen tuntitehtävissä menestymisessä, oppilaat voivat siirtyä nopeammin ideoimaan käsityöprojektia.

Arviointi

Opettaja arvioi ryhmän Micro:bitin ja visuaalisen ohjelmointiympäristön osaamista sekä valmiutta siirtyä käsityöprojektin ideointiin tuntitehtävissä menestymisen pohjalta. Arvio perustuu havainnointiin ja oppilaiden kanssa käytyihin keskusteluihin. Oppilaat reflektoivat osaamistaan ryhmän (yksilö- ja vertaisarviointi) ja opettajan kanssa käymissään keskusteluissa.

Sisältö

Opetukseen sisältyy tässä kokonaisuudessa sekä yksilöllistä että ryhmässä opiskelua. Oppilaat pääsevät kokonaisuuden aikana osallistumaan opetukseen keskustelujen ja oman työpanoksen kautta. Ryhmätyöskentelyssä oppilaat pohtivat ohjelmoinnin merkitystä koulupäivissään ja pääsevät ideoimaan yhdessä vertaisten kanssa ohjelmoinnin jatkohyödyntämistä koulupäivien sujuvuuden parantamiseksi.

Aikataulutus, opetusmenetelmät, materiaalit, oppimisympäristöt

Ajankäyttö	Sisältö	Opetusmenetelmät	Työskentelymateriaalit ja -välineet, fyysinen oppimisympäristö	Valmistautuminen opetukseen
n. 15 min	Keskustellaan oppilaiden kanssa ohjelmoinnista. Esim. • Mitä ohjelmointi on? • Missä ohjelmointia ympäristössä on? Millaisia toimintoja ohjelmoinnilla on toteutettu? • Oppilaiden ohjelmointikokemuksen kartoittaminen	Yhteinen keskustelu	ATK- luokka tms. tila, jossa on tietokone jokaiselle 2-3 oppilaan ryhmälle (tässä vaiheessa koneita ei vielä avata)	ATK-luokan tai vastaavan varaaminen
n. 20 min	Micro:bit ja visuaalinen ohjelmointiympäristö	Opettajavetoinen esittely Micro:bit -minikontrollerista ja visuaalisesta ohjelmointiympäristöstä (MakeCode)	Opettajan tietokone Dokumenttikamera, Valkokangas ja projektori tai vastaava Micro:bit -aloituspakkaus Opetusmateriaali Oppilaiden tietokoneet ovat tässä vaiheessa vielä kiinni	Opetusmateriaalin valmistelu (esim. liite 1)
n. 10 min	Ryhmäjako	Opettaja jakaa oppilaat 2-3 hengen ryhmiin		Alustava ryhmäjako (mahdollisia muutoksia alun keskustelun perusteella, koska ohjelmointia osaavat jaetaan eri ryhmiin mahdollisuuksien mukaan)
Välitunti				
n. 5 min	Tehtävien antaminen oppilaille	Opettaja kertoo tuntitehtävät oppilaille ja vastaa mahdollisiin kysymyksiin		Tuntitehtävien ja käsityöprojektitehtävän valmistelu Tuntitehtävien kirjoittaminen näkyville
n. 25 min	Tuntitehtävät	Oppilaat työskentelevät 2-3 hengen ryhmissä ja siirtyvät projektitehtävän ideointiin opettajan luvalla	Tietokoneet, paperia ja kyniä	
n. 15 min	Käsityörojektin ideointi	Oppilaat jatkavat työskentelyä samoissa 2-3 hengen ryhmissä	Paperia, kyniä, muita muistiinpanotarvikkeita	Käsityöprojektin ideointitehtävän kirjoittaminen näkyville tuntitehtävien aikana

Tuntitehtävät

1. Keksikää ryhmällenne nimi ja tehkää Micro:bitistä ryhmälle nimilappu.
2. Ohjelmoikaa Micro:bit toimimaan noppana.
3. Käyttäkää Micro:bitin ledien ohjelmointia ja tehkää vähintään viiden erilaisen merkin toistuva sarja.

Tuntitehtävissä saa käyttää hyödyksi MakeCode -ohjelman ohjemateriaaleja

Käsityöprojektin ideointitehtävä

Pohtikaa koulupäiväänne ja kertokaa ryhmälle, millaisiin epäkohtiin ohjelmoinnilla on jo vaikutettu ja mitä asioita sen avulla vielä voisi parantaa. Kirjatkaa ylös kaikki ideat, vaikka ne vaikuttaisivat mahdottomiltakin.

Huomioita

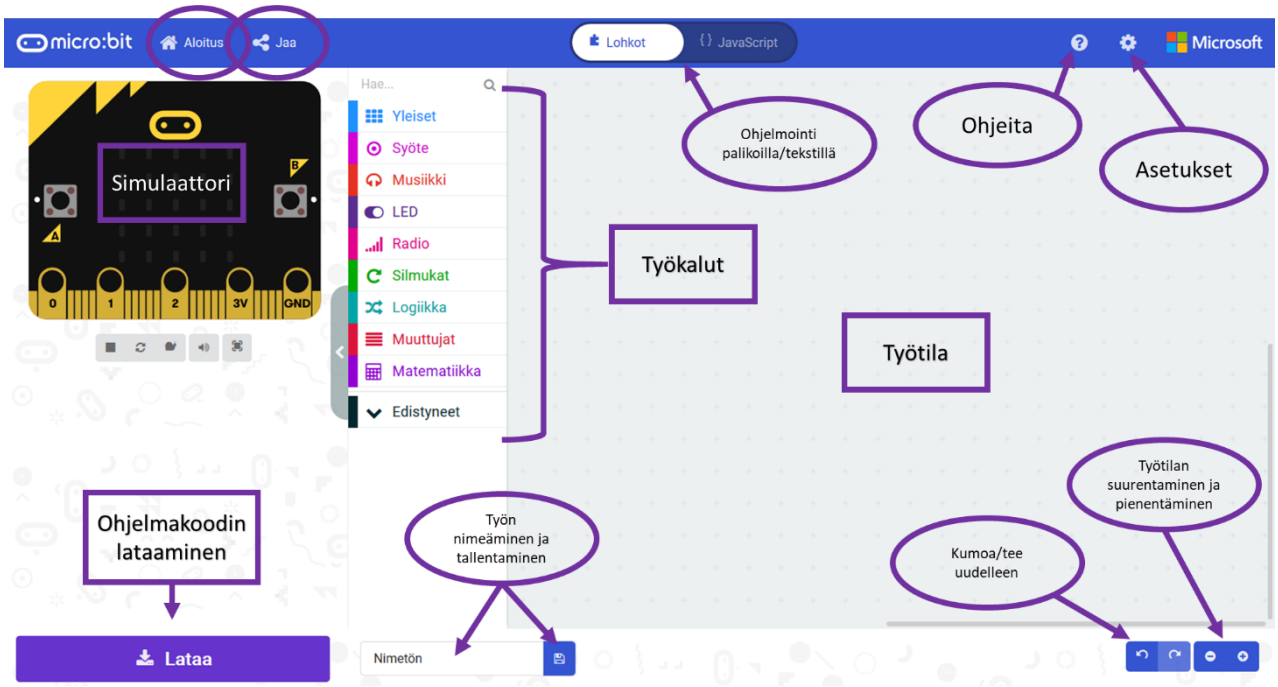
Tämä kokonaisuus ottaa huomioon ohjelmoinnin ”kaikkiallisuuden”. Ohjelmoinnin esiintymisestä arkipäivän ympäristöissä keskustellaan jo oppikokonaisuuden alussa. Lisäksi aiheeseen syvennytään vielä jatkossakin ryhmätyöprojektissa, jossa opittua hyödynnetään ja sovelletaan ryhmän tuottamien ideoiden ja suunnitelman pohjalta. Ryhmissä työskentelyn taustalla on harjoituttaa oppilaiden yhteistyötaitoja ja erilaisten oppilaiden vahvuuksien hyödyntämistä riippumatta ryhmän kokoonpanosta. Siksi opettaja tekee ryhmäjaon.

Lähteet

Lyman, C. (2019). *Coding & Innovation using Micro:bits*. Viitattu: 24.11.2019. Saatavissa: https://drive.google.com/file/d/18qPOv7s5-moqmhoU5caBzy-9UBLI2Hk-/view.

Liitteet

Liite 1. MakeCode -ohjelmointiympäristön esittelydia



Ohjelmointiympäristön yleisesittely Lymania (2019) mukaillen.

Kertolaskupeli Microbitilla, 7.lk

Jani Heinonen, CC BY-SA 4.0

6.1.2020

Kertolaskupeli Microbitilla

Tarkoitus on hyödyntää ohjelmointia matematiikan tunnilla, koska se on osa matematiikan opetussuunnitelmaa. Lisäksi ohjelmoimalla saadaan peli, jota voi hyödyntää uuden asian opiskelussa. Ohjelman avulla voi varmentaa esimerkiksi matematiikan merkkisääntöjä, jotka ovat haastavia monille aluksi.

Tavoitteena on, että tämän projektin avulla oppilaat saavat käsityksen ohjelmoinnin hyödyistä matematiikassa. Samalla tutustutaan robotiikkaan Microbit-piirin avulla. Opettaja arvioi työskentelyä, joka sisältää sekä ohjelmoinnin, että matematiikan. Tämä muodostaa yhden kokonaisuuden. Oppilaat arvioivat toistensa pelejä ja näin oppilaat saavat vertaisarviointia toisiltaan. Myös oman pelin arvioiminen on tärkeää.

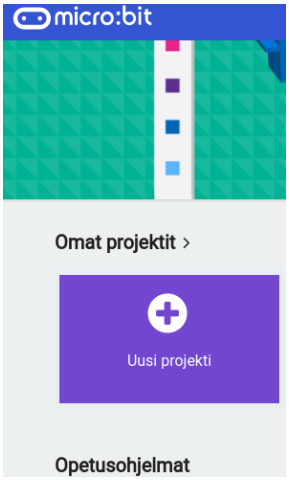
Välineinä käytetään Microbit-piirejä. Johdantomateriaaliksi annetaan kuvaus pelin ideasta. Osa voi päästä sen avulla liikkeelle ja lähteä parantelemaan annettua ohjeistusta. Lopun ryhmän kanssa toteutetaan yhdessä malliratkaisu, jota on tarkoitus muokata myöhemmin omanlaiseksi. Oppilaat motivoituvat tästä aiheesta, koska lopputuloksena on peli, jossa voi kilpailla luokkalaisten kanssa. Yhteistyöhön rohkaistaan pelin ja sen yhteydessä tulevan vertaisarvioinnin myötä.

Ohjelmoinnin aikana oppilaat huomaavat, että samaa koodia hieman muokkaamalla on mahdollista tehdä monenlaisia ohjelmia. Tämän oivaltamalla he voivat innostua koodaamaan ohjelmia muissakin tilanteissa.

Pelin tarkoitus on harjoitella Microbitilla ohjelmointi 7.luokan negatiivisten ja positiivisten lukujen tulon yhteydessä.
Pohjatietoja ohjelmointiin ei tarvita. Projekti sopii hyvin ensimmäiseksi ohjelmointityöksi.
Kesto 45-90 minuuttia riippuen aiemmasta kokemuksesta ja opettajan antamista vinkeistä.

Ohjeet:
Avaa sivusto: https://makecode.microbit.org

Valitse Uusi projekti

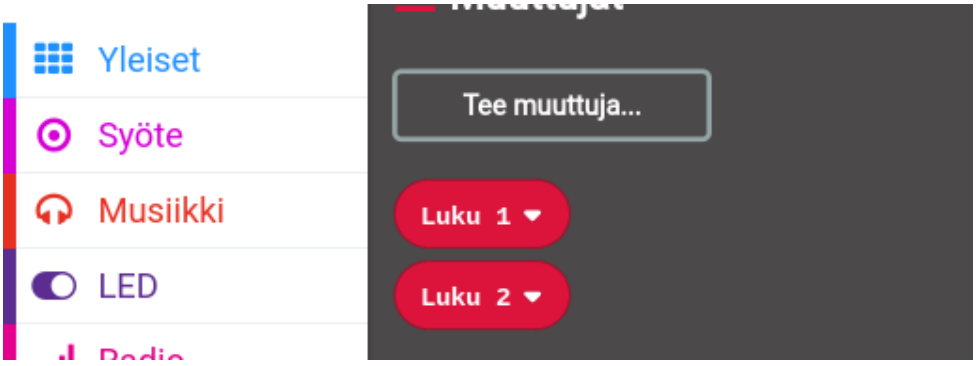


Pelin idea on, että se valitsee satunnaisesti kaksi lukua, jotka se näyttää pelaajalle. Tämän jälkeen peli laskee lukujen tulon. Oppilaan on tarkoitus kertoa vastaus ennen kuin peli tekee sen. Jos vastaus on oikein. Hän voi merkitä paperille pisteen.

Ohjelmointi:

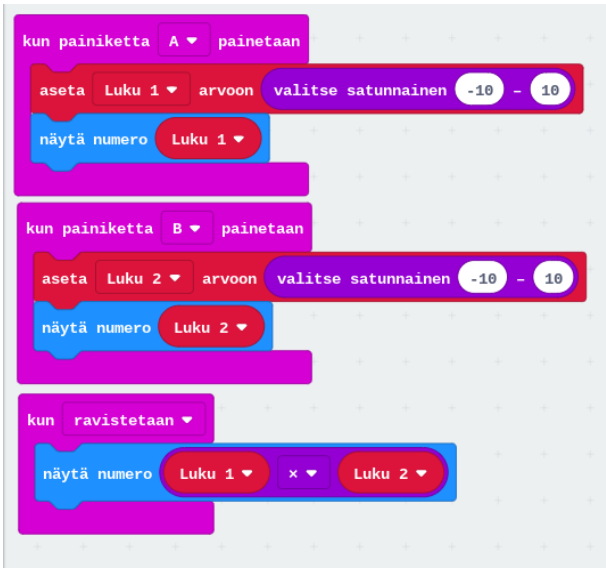
Kannattaa antaa oppilaiden ensin yrittää pohtia tehtävää ja kokeilla itse, miten pelin voisi toteuttaa. Alla on yksi tapa, jota voi ehdottaa oppilaille.

Tee muuttujat: Luku 1 ja Luku 2. (Löytyy Muuttujat-valikon alta)



Sen jälkeen tulee tehdä 3 erilaista lohkoa.

1. lohko käynnistyy painamalla A-näppäintä. Tällöin ohjelma tallentaa muuttujan arvoksi satunnaisen luvun näyttää sen (Luku 1).
2. lohko käynnstyy painamalla B-näppäintä. Tällöin ohjelma tallentaa muuttujan arvoksi satunnaisen luvun ja näyttää sen (Luku 2).
3. lohko käynnistyy ravistamalla piirilevyä. Tällöin ohjelma laskee Luku 1:n ja Luku 2:n tulon ja näyttää sen.



Nyt ohjelma on valmis ja se tulee siirtää Microbitiin.

- Kytke piirilevy koneeseen ja paina näytöllä näkyvää Tallenna-kuvaketta. Tallenna ohjelma ulkoiselle levylle, jonka nimi on Microbit. Hetken päästä ohjelmaa voi käyttää Microbitillä ilman tietokonetta.

Kun ohjelma on saatu tehtyä ja sillä on pelattu, niin sitten on tarkoitus aloittaa oman pelin kehittäminen.

Kysymyksiä oppilaille:
Miten voisit muuttaa pelin laskemaan jotain muita laskutoimituksia?
Miten voisit muuttaa satunnaisten lukujen arvoja?
Voisiko edellisen esimerkin tehtävän tehdä jotenkin eri tavalla?

Rakenna erilaisia pelejä ja käy esittelemässä niitä muille. Kysy myös pariltasi, miten peliäsi voisi kehittää.

Laskutoimituksia luvuilla, 7.lk

Aija Elo, CC BY-SA 4.0

27.10.2020

Aihepiirin valinta ja rajaus

Aine on matematiikka, siinä laskutoimitukset luvuilla, aivan alkeita. Tämä koska asia on ajankohtainen seiskojen matematiikan kurssillani, mutta se toteutetaan vasta myöhemmin. Toivon, että saan siitä kehitetyksi vielä projektinkin.

Aiheena ovat aivan Pythonin perusteet. Ne on sijoitettu lukujen ja laskutoimitusten joukkoon, peruslaskutoimitusten harjoitteluun. S1 ajattelun taidoissa on mukana algoritmisen ajattelun kehittäminen, samoin ohjelmoimisen ja hyvien ohjelmointimenetelmien harjoittelu. Samoin mainitaan itse tehtyjen ja valmiiden ohjelmien käyttö. Työskentelyn taidoissa harjoitellaan tieto- ja viestintätekniikkaa matematiikassa.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppimistavoitteena on perussyöttö ja tulostus sekä niiden ohjaaminen koodilla, muuttuja ja niiden tyypit, lukujen vertailu ja vertailusilmukan rakentaminen

Arviointiin voidaan huomioida onnistuneiden tehtävien määrä, samoin erilaista itsearviointia ja vertaisarviointia. Tehtäviä voidaan laatia joko yksittäin, mutta ryhmässä tekeminen voi olla hyvä ratkaisu. Kun oppilaat voivat tehdä ryhmässä ja tärkeintä on saada onnistumisia kokoon, ei numeerinen arviointi ehkä ole niinkään tärkeää.

Tämä pitää paikkansa erityisesti, jos kokonaisuuteen käytetään aikaa, jolloin arviointi on jo pitkälti valmis ja tarkoitus on silti jatkaa mielekästä opiskelua. Tämä sopii erityisesti, jos kokonaisuudesta saa kasvatetuksi myös pidemmän kokonaisuuden, johon voi lisätä pelillisiä ja kilpailullisia elementtejä.

Tässä tapauksessa kahden tunnin kokonaisuus antaa perusohjelmointivalmiuksia, joita voidaan hyödyntää jatkossa.

Työskentelyvälineet ja opetusmenetelmät

Materiaalina voidaan käyttää oppikirjan sisältöä ja sen lisämateriaalin linkkejä, kuten https://www.ohjelmointiputka.net/oppaat/. Sen avulla voidaan suorittaa pikkutehtäviä, vastaavasti kuin https://tie.koodariksi.fi/ -sivustoa. Ohjelmointiputka sisältää paljon valmiita ohjelmointi-ideoita samoin kuin jopa Pythonin lataussivun pitäisi löytyä ohjeineen kohdasta oppaat.

Johdantomateriaali löytyy siis oppikirjasta ja ohjelmistot tarjoavat harjoittelumahdollisuuden. Oppimiseen voi yhdistää erilaista pelillistä toimintaa, joka lisää oppilaiden motivointia ja kannustaa ryhmätyöhön (varsinkin pidemmälle ehdittäessä). Oppilaat voivat ryhmissä etsiä kiinnostavia tehtäviä ja äänestää niistä kiinnostavina pitämiään (vertaa Kahoot). Tunteja voidaan silloin jaksottaa osioilla, joissa oppilaat työskentelevät ohjelmoiden ja etsien pienemmissä ryhmissä. Välillä ryhmä työskentelee kaikki yhdessä, jolloin tuloksia vertaillaan.

On myös mahdollista, että oppilaat voivat esitellä aiempaa osaamistaan jollakin muulla menetelmällä, tällaisen yhteisjakson aikana.

Lähteet:

Oppikirja: Kuutio 7 Luvut ja laskutoimitukset. Sanoma Pro.

https://www.oph.fi/fi/koulutus-ja-tutkinnot/perusopetuksen-opetussuunnitelman-perusteet

Ohjelmoinnin oppikokonaisuus, 7.lk

Hanna-Leena Hämäläinen, CC BY-SA 4.0

28.11.2020

Ohjelmoinnin oppikokonaisuus 7.luokkalaisille

Aihepiirin valinta ja rajaus

- Suunnittelin kahden oppitunnin mittaisen ohjelmoinnin oppikokonaisuuden matematiikan tunnille 7.luokkalaisille.
- Tutustutaan ohjelmointiin yleisesti, harjoitellaan antamaan täsmällisiä komentoja sekä opitaan tulostamaan tekstiä ja tekemään peruslaskutoimituksia Python-ohjelmointikielellä.

Oppimistavoitteiden määrittely ja arviointi

- Tavoitteena on huomata, että ohjelmoinnissa komentojen tulee olla tarkkoja. Tavoitteena on myös tutustua Python-ohjelmointikieleen ja oppia käyttämään print-komentoa tekstin tulostukseen ja peruslaskutoimitusten tekoon.
- Opettaja kiertää luokassa ohjaten oppilaita eteenpäin tehtävissä ja havainnoi oppilaiden edistymistä ohjelmoinnissa.
- Oppilas tekee lopuksi itsearvioinnin.

Työskentelyvälineet ja opetusmenetelmät

- Oppikokonaisuudessa käytetään ensimmäisellä tunnilla vain paperia ja kynää. Toisella tunnilla otetaan Chromebookit käyttöön.
- Aluksi käydään yhdessä läpi, mitä ohjelmointi on ja kuinka tarkat komennot pitää olla. Sen jälkeen tutustutaan Python-ohjelmointikieleen tekemällä yksinkertaisia print-komentoja selainpohjaisen python online-tulkin avulla.
- Ensimmäisellä tunnilla teemme hauskan parityön, jossa harjoitellaan komentojen antoa. Se innostaa ja motivoi oppilaita ohjelmoinnin pariin.
- Oppilaat istuvat pareittain, joka rohkaisee miettimään tehtäviä yhdessä kaverin kanssa. Opettaja kiertää luokassa kannustaen ja ohjaa oppilaita tehtävissä eteenpäin.

TUNTISUUNNITELMA:

1. **oppitunnin sisältö:**

   Käydään aluksi muutaman dian kanssa läpi, mitä ohjelmointi on. Valitaan dioihin kiinnostavia kuvia.

   -Ohjelmointi on… tietokoneelle tai vastaavalle laitteelle jollakin tavalla annettavia tarkkoja toimintaohjeita.

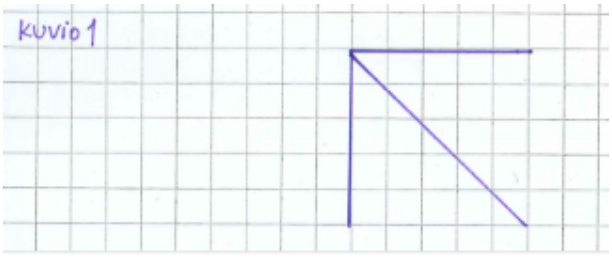
-Toimintaohje eli *algoritmi* täytyy kuvata jollakin tietokoneen ymmärtämällä *ohjelmointikielellä*.
-Ohjelmointi vaatii tarkkuutta ja huolellisuutta, koska tietokone noudattaa saamiaan ohjeita mekaanisesti.
-Pienikin virhe ohjelmassa voi aiheuttaa täysin odottamattoman lopputuloksen.
-Oikein laaditun ohjelman tietokone suorittaa erittäin nopeasti ja virheettömästi.
-Ohjelmoinnin oppiminen vaatii paljon aikaa.
-Ohjelmointia oppii parhaiten käytännön kautta tekemällä paljon pieniä harjoitusohjelmia.

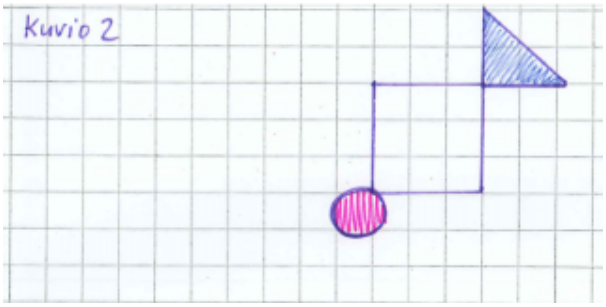
Sen jälkeen keskustellaan seuraavista toimintaohjeista, onko ohjeet tarpeeksi tarkat esimerkiksi robottia ajatellen:

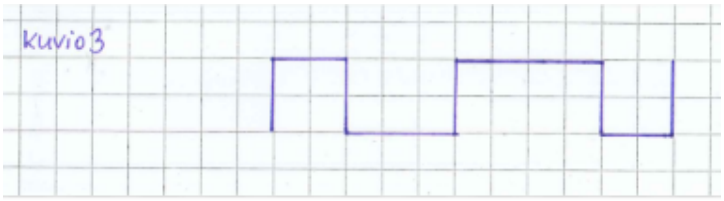
1.Kuumenna maito tai vesi kiehuvaksi paksupohjaisessa kattilassa.
2.Lisää joukkoon kaurahiutaleet ja ripaus suolaa.
3.Keitä puuroa samalla sekoitellen noin 5 minuuttia.
4.Ota kattila pois levyltä ja anna puuron hautua muutaman minuutin ajan.
5.Tarjoile puuro marjojen, hedelmien tai marjakeiton kera.

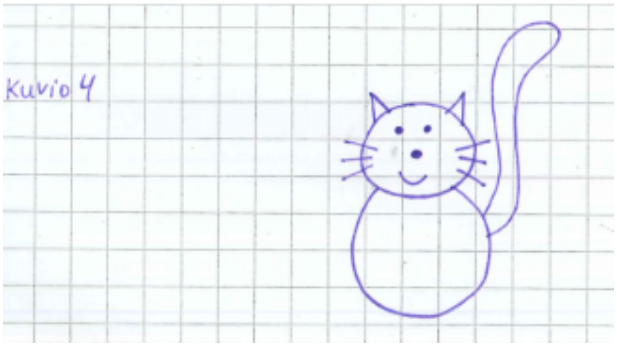
Huomataan, että ohjeiden pitäisi olla tarkemmat. Esimerkiksi ainesten tarkat määrät puuttuvat.

Seuraavaksi tehdään hauska pariharjoitus. Toinen katsoo taululta kuvaa ja antaa parille mahdollisimman tarkat ohjeet piirtää paperille samanlainen kuvio. Ohjeiden antaja ei saa katsoa parin aikaansaannosta ohjeita annettaessa, eikä piirtäjä saa katsoa taululle. Kun ensimmäinen kuva on valmis, he vertaavat, oliko ohjeet olleet tarpeeksi tarkat. Vaihdetaan piirtäjää. Molemmat pääsevät antamaan kaksi kertaa ohjeet ja piirtämään kaksi kertaa. Alla kuvat.









2. **oppitunnin sisältö:**

Toisella tunnilla otetaan Chromebookit mukaan. Jokaisella on oma Chromebook, mutta oppilaat istuvat pareittain. Käytetään tehtäviin alla olevaa Python online-tulkkia.
https://www.jdoodle.com/python3-programming-online/

Ohjeistetaan oppilaita:
Valkoiseen laatikkoon kirjoitetaan koodi, jonka jälkeen painetaan sinistä "Execute"-painiketta (eli suorita). Mustaan laatikkoon alle tulee ohjelman tulostus näkyviin.

Oppilaat saavat monisteen, jossa on alla olevat ohjeistukset ja tehtävät.

1. Teksti tulostetaan komennolla print( ).
Tulostettava teksti laitetaan lainausmerkkien sisään.

Kirjoita ohjelmaan:

print(”Jippikaijee, tämähän tulostuu!”)

2. Ohjelmaan voidaan lisätä kommentteja, jotka auttavat ohjelman seuraamisessa ja läpikäymisessä.

Kommentti aloitetaan #-merkillä.
Kirjoita ohjelmaan:

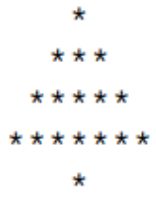
print(”Opimme ohjelmoimaan Pythonilla.”)

#Tämä on kommenttirivi, jota ei tulkata.

3. Kirjoita ohjelma, joka tulostaa nimesi ruudulle.

4. Kirjoita ohjelma, jonka tulostus on seuraava:

5. Kirjoita ohjelma, jonka tulostus on seuraava:



6. Print-komennon sisään voi laittaa laskutoimintoja. Silloin tulostuu pelkkä laskun vastaus.

Huomioi, ettei nyt laiteta lainausmerkkejä.

Kirjoita ohjelmaan:

```
print(3+1)
print(4*5)
print(10-2)
print(12/3)
```

7. Kirjoita ohjelma, joka tulostaa seuraavien laskutoimitusten vastauksen.

a) 12-2*9

b) 126 / (30-24)

Python Turtle ja monikulmiot, 7.lk

Tomi Hautakangas, CC BY-SA 4.0

10.4.2021

LUMATIKKA 3: Algoritmisen ajattelun kehittäminen (KEVÄT 2021)

TUNTISUUNNITELMA JA OPETTAJAN OHJE

Matematiikka 7. luokka - Monikulmiot, kaksi oppituntia (2 * 45 min)

Tämä on kahden oppitunnin tuntisuunnitelma 7. luokan matematiikan tasogeometriaan monikulmioiden opiskeluun. Suunnitelma on laadittu samalla opettajan ohjeeksi.

AIHE:

Opiskelemme säännöllisten monikulmioiden piirtämistä ja ominaisuuksia Pythonin Turtle-grafiikkaa käyttäen.

TYÖVÄLINEET:

Oppilailla on henkilökohtaiset Chromebookit ja Google-tunnukset. Teemme tehtäviä Chrome-selaimessa käytettävällä Python-ohjelmistolla Trinket. Oppilaat kirjautuvat Trinket-palvelimelle omilla Google-tunnuksillaan. Tehtävät voi toki tehdä missä tahansa Python-ohjelmointiin tarkoitetussa ympäristössä.

Oppilaan ohje jaetaan oppilaille Google Classroomissa tehtävänä. Oppilaat kopioivat Python-tehtävien koodit suoraan Trinket-palvelimelta Classroomissa jaettuun tehtävään. Oppilaiden tarvitsemat ohjelmakoodit koodit sisältyvät Classroomissa jaettaviin tehtäviin.

Tämän tuntisuunnitelman lopussa on tehtävien ratkaisut eli tehtävien esimerkkikoodit opettajaa varten. Oppilaille jaettavia koodeja voidaan muokata vielä sen mukaan, mikä on oppilaan taitotaso. Taitavimmat voivat saada vähemmän valmista koodia.

Sekä oppilaiden Classroom-tehtävät että ohjelmakoodit jaetaan sijaiselle ennen oppituntia.

ESITIEDOT JA KERTAUS:

Oppilaat osaavat kopioida ja liittää tekstiä ja linkkejä Classroomissa jaettuihin tehtäviin ja palauttaa tehtävät.

Oppilaat ovat aiemmin tutustuneet Python-ohjelmointiin ja Turtle-grafiikkaan ennen tätä tuntia ja osaavat kirjautua Trinket-ympäristöön. Olemme piirtäneet suorakulmioita ja omia vapaamuotoisia kuvioita aiemmalla kerralla, joten yksinkertaiset Python Turtlen komennot ovat osittain tuttuja.

Oppilaat tietävät, että suorakulmioissa on neljä suoraa kulmaa (90°). Oppilaat osaavat piirtää käsin kynällä ja viivaimella suorakulmioita ja mitata suorakulmioiden sivujen pituuksia sekä laskea suorakulmion piirin.

Oppilaat tekevät ensin yhden tehtävän, jossa piirretään neliö ilman toistorakennetta. Tämä on heille aiemmalta kerralta tuttu tehtävä mutta toistorakenne on täysin uusi asia. Opettelemme for-silmukan.

OPPIMISTAVOITTEET:

Oppimistavoitteena on useita. Tarkoitus on oppia lisää Python-ohjelmointia ja kerrata Turtle-moduulin käyttöä. Tarkoitus on monipuolistaa ohjelmointitaitoja ja oppia uusi käsite eli toistorakenne. Tarkoitus on myös kerrata tasogeometriaan ja tasokuvioihin liittyvät käsitteet sivu, sivun pituus ja kulma sekä käyttää näitä ohjelmoitaessa. Tarkoitus on oppia piirtämään säännöllisiä monikulmioita ja laskemaan säännöllisen monikulmion kulman suuruus.

Tarkoitus on oppia, että aiemmin tehtyä koodia voi hyödyntää seuraavissa tehtävissä eikä useinkaan kannata kokonaan kirjoittaa alusta loppuun yksittäiseen tehtävään tarvittavaa koodia. Lisäksi on tarkoitus oppia kommentointia. Usein oppilaat eivät lainkaan halua

kommentoida omaa koodiaan. Nyt kirjoitetaan kaikki kommentit ja niistä saa osan pisteistä, kun tehtäväpalautukset arvioidaan. Tämä tehdään nimenomaan siksi, että ymmärrettäisiin, mitä missäkin kohdassa ohjelmakoodia tapahtuu.

Kahden oppitunnin välinen raja ei ole tarkka. Ensimmäisellä tunnilla tehdään tehtävät, joissa piirretään tavallisia monikulmioita ja opitaan toistorakenne. Toisella tunnilla asioita kerrataan, piirretään n-kulmioita sekä piirretään kokonaan omia erilaisista säännöllisistä monikulmioista muodostuvia kuvioita.

Nopeimmat oppilaat voivat edetä seuraaviin tehtäviin itsenäisesti.

ARVIOINTI

Jokainen oppilas tekee kaikki ensimmäisen oppitunnin tehtävät ja palauttaa ne. Hyvä suoritus ensimmäisellä tunnilla on, että saa kaikki 1-6 tehtävää tehtyä ja kirjoitettua kommentteja. Toisella tunnilla hyvä suoritus on, että oppii pohtimalla ja kokeilemalla, miten aiemmalla tunnilla opetetut asiat voidaan yleistää koskemaan myös n-kulmioita ja saa piirrettyä jonkin monikulmioista koostuvan kuvan, logon tai vastaavan.

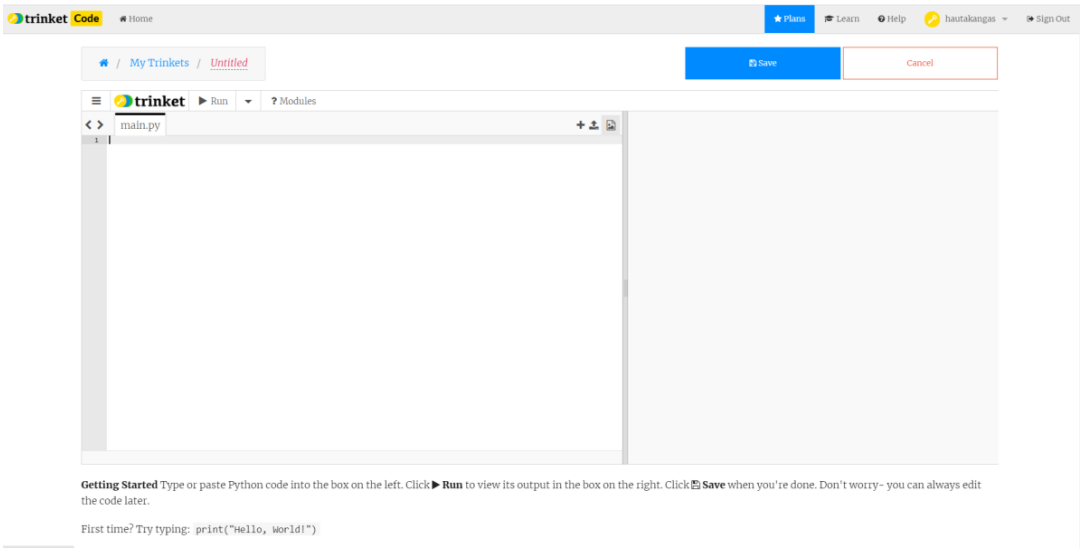
TYÖSKENTELY

Oppilaat avaavat Trinket-ympäristön ja kopioivat heille Classroomissa jaetut koodit suoraan Trinketin ohjelmointi-ikkunaan ja tekevät tehtävät.

LAINAUS OPPILAAN OHJEESTA:

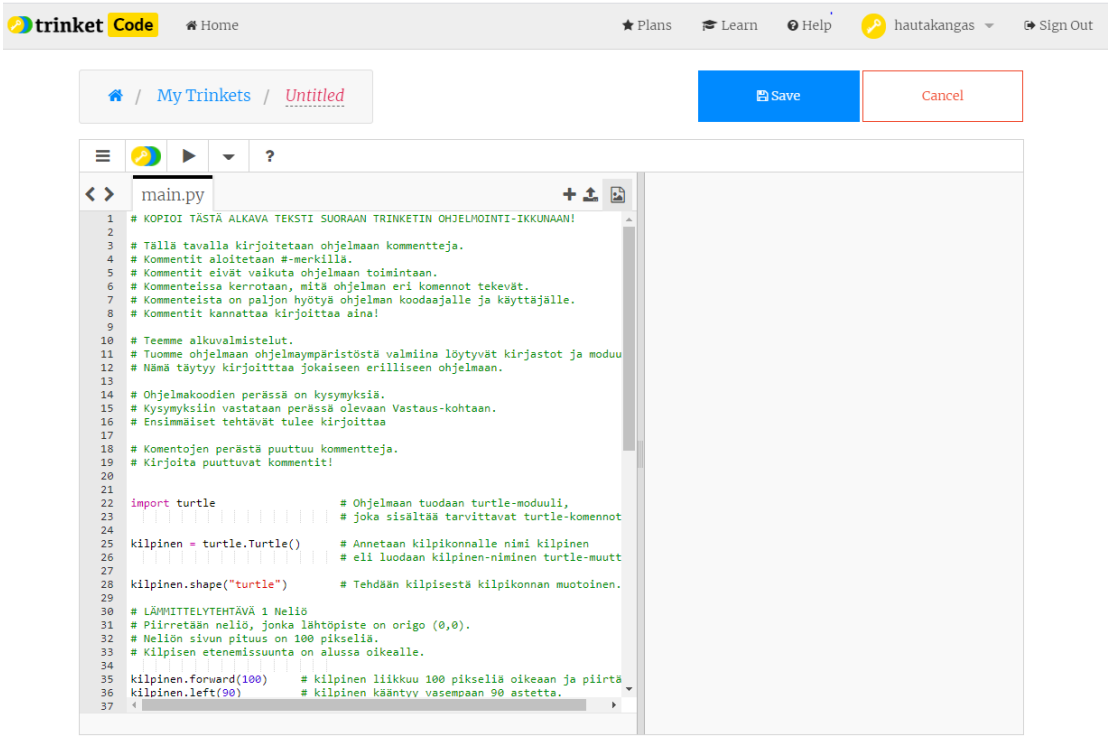
1. Mene Chrome-selaimesta tai tästä linkistä Trinket-palvelimelle.
2. Kirjaudu Trinket:iin omilla Google-tunnuksillasi.
3. Klikkaa käyttäjätunnustasi: Aukeaa valikko.
4. Valitse New Trinket ja Python.

Aukeaa tämä:



Aloita ohjelmointi. Kopioi vasemmanpuoleiseen ohjelmointi-ikkunaan Tehtävän 1 koodi Classroomissa jaetusta materiaalista TEHTÄVÄSTÄ 1.

TEHTÄVÄ 1. NELIÖ Ikkunan pitäisi näyttää kopioinnin jälkeen tältä:



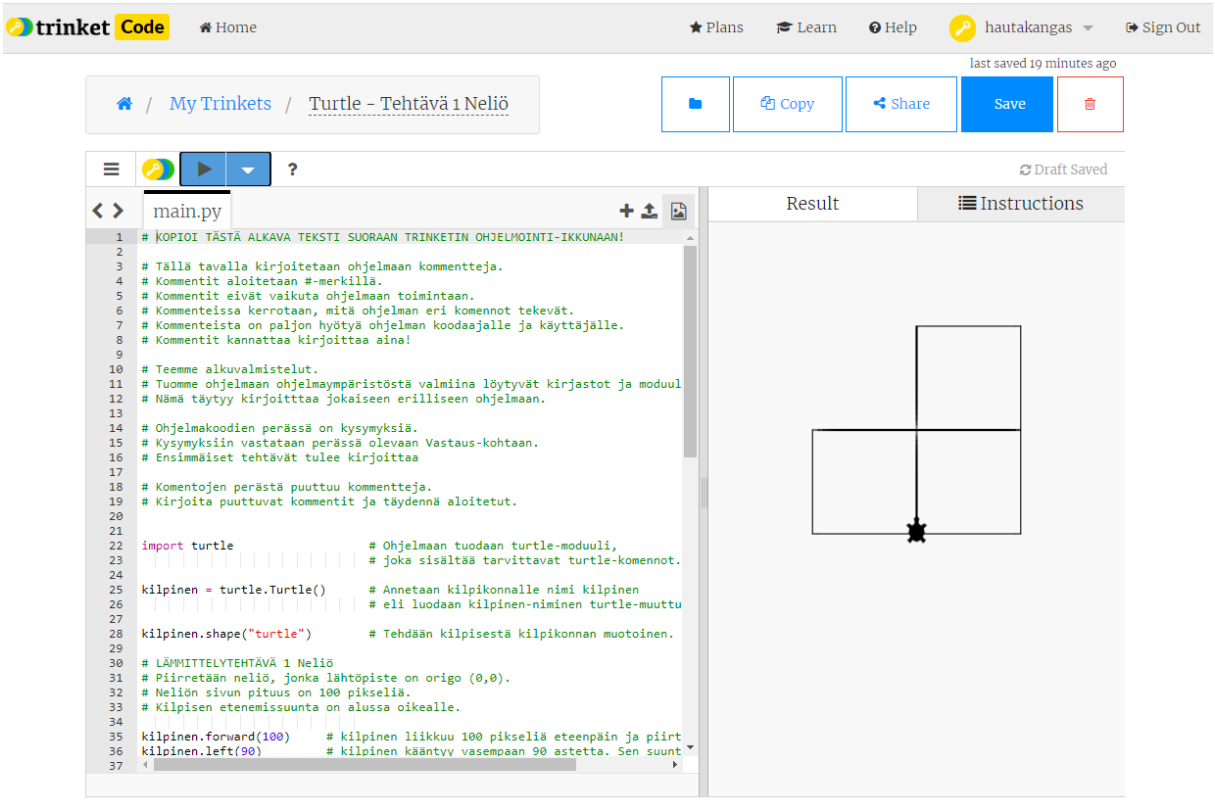
TEHTÄVÄ 1. jatkuu ...

Tee koodiin tarvittavat muutokset ja suorita ohjelma.

Vastaa kommenttien kysymyksiin.

Alla olevassa kuvassa ohjelman suoritus on keskeytetty juuri ennen loppua eli tämän tapaista sinun pitäisi saada, kun suoritat ohjelman sen jälkeen, kun olet tehnyt siihen ensin tarvittavat muutokset.

Jos saat jotain ihan muuta, mieti mihin suuntaan kilpinen kääntyy ja etenee.



Kopioi valmis koodi takaisin Classroom-tehtävään.

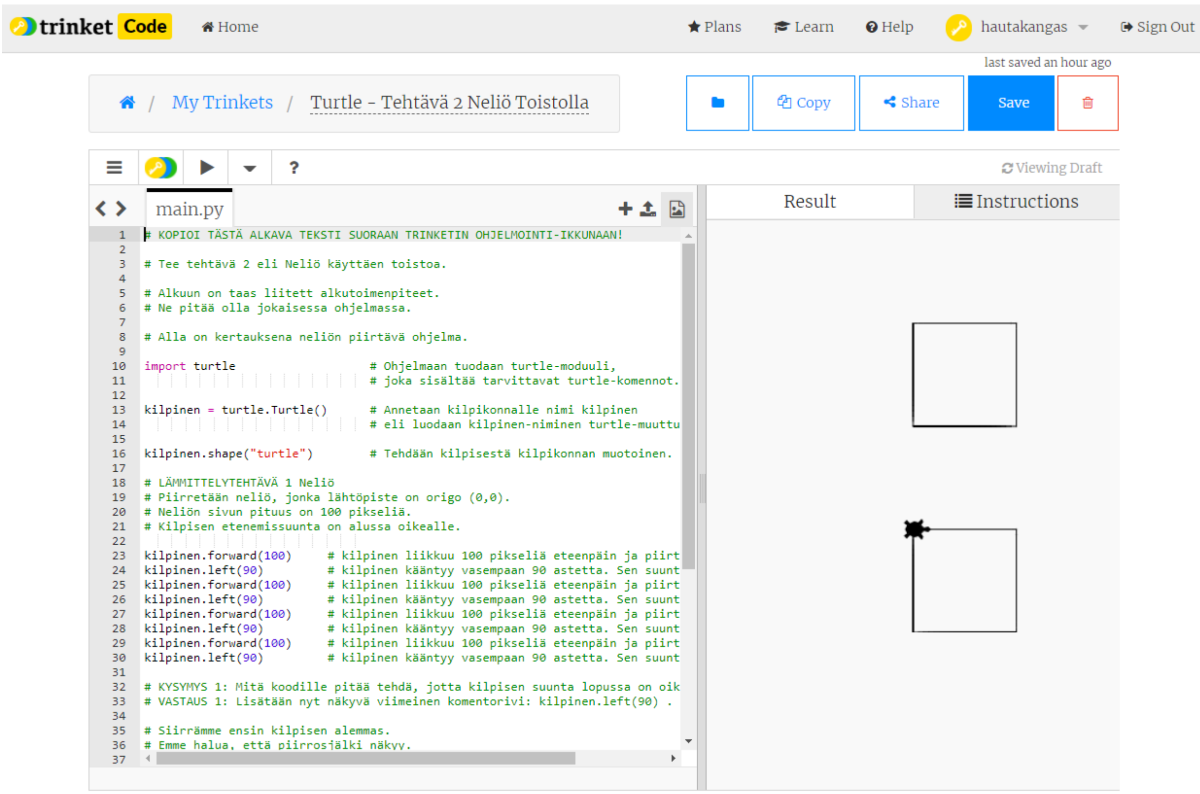
Opettajan saama valmis tehtävän koodi on erilainen kuin oppilaalle jaettava, joten voi kysyä häneltä apua, jos jäät jumiin. Myös kaveri voi auttaa.

TEHTÄVÄ 2

Avaa uusi tyhjä ohjelma Trinketissä ja kopioi siihen koodi Classroom-tehtävästä TEHTÄVÄ 2.

Tehtävän koko ohje sisältyy ohjelmakoodin kommentteihin.

Suoritetun ohjelman kuva on tässä:



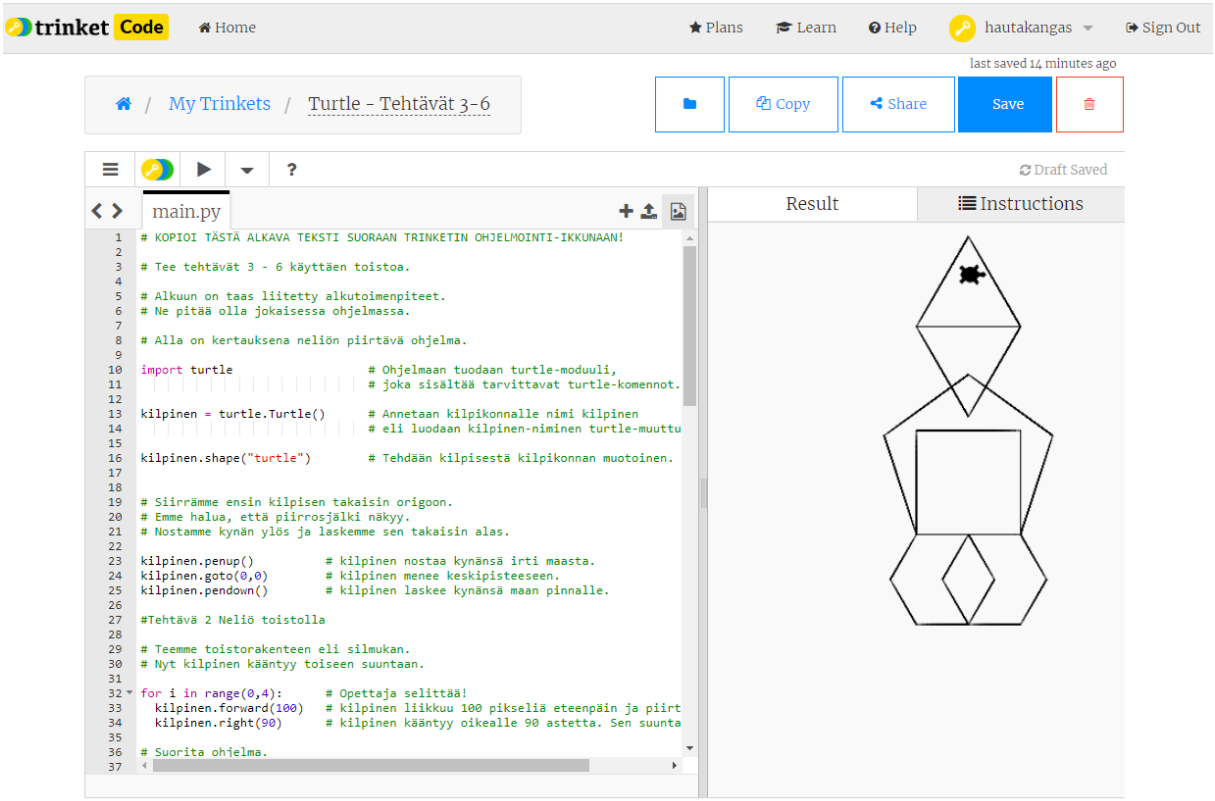
Koodin opettajan versio on tämän suunnitelman lopussa.

TEHTÄVÄT 3- 6

Avaa uusi tyhjä ohjelma Trinketissä ja kopioi siihen koodi Classroom-tehtävästä TEHTÄVÄT 3-6.

Tehtävän koko ohje sisältyy ohjelmakoodin kommentteihin. Koodi on tämän suunnitelman lopussa.

Suoritetun ohjelman kuva on tässä:

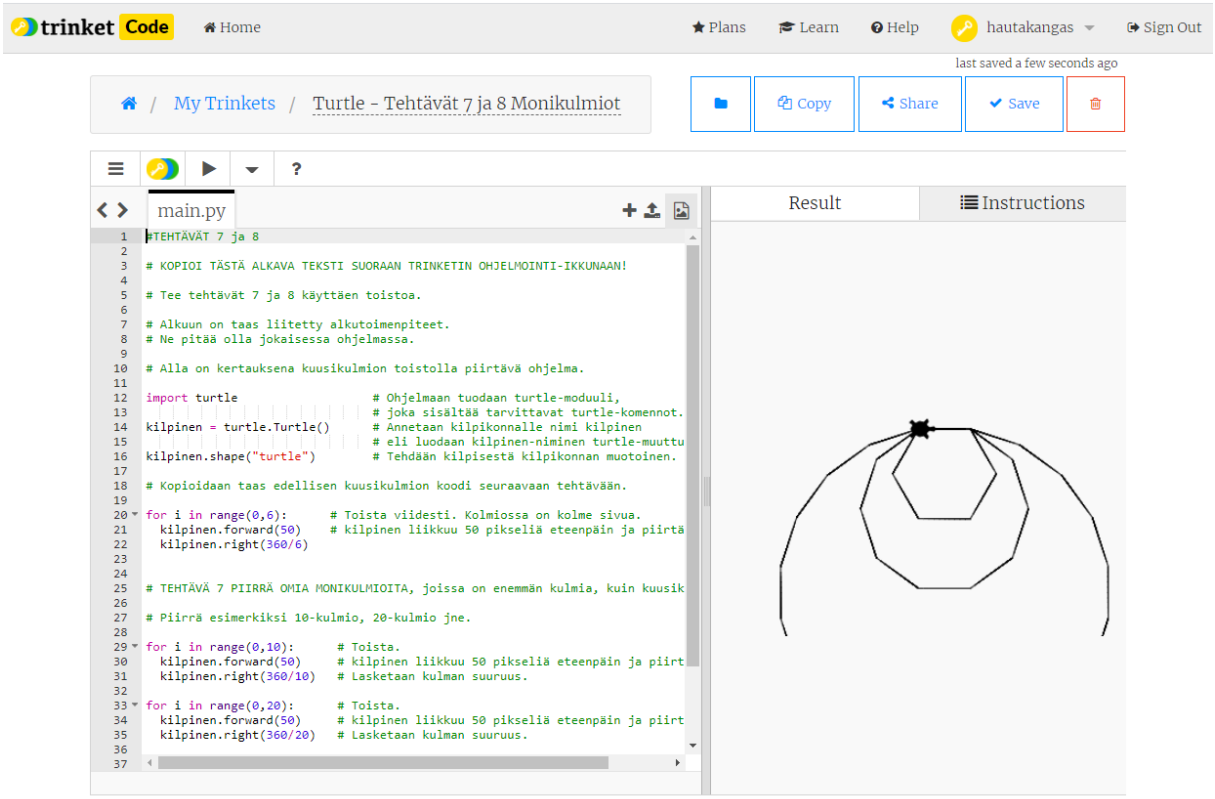


TEHTÄVÄT 7 ja 8

Avaa uusi tyhjä ohjelma Trinketissä ja kopioi siihen koodi Classroom-tehtävästä TEHTÄVÄT 7 ja 8.

Tehtävän koko ohje sisältyy ohjelmakoodin kommentteihin. Koodi on tämän suunnitelman lopussa.

Suoritetun ohjelman ja hyvin yksinkertaisen logon kuva voisi olla tällainen:



KOODIT OPETTAJALLE:

TEHTÄVÄ 1

#TEHTÄVÄ 1

KOPIOI TÄSTÄ ALKAVA TEKSTI SUORAAN TRINKETIN OHJELMOINTI-IKKUNAAN!

Tällä tavalla kirjoitetaan ohjelmaan kommentteja.
Kommentit aloitetaan #-merkillä.
Kommentit eivät vaikuta ohjelmaan toimintaan.
Kommenteissa kerrotaan, mitä ohjelman eri komennot tekevät.
Kommenteista on paljon hyötyä ohjelman koodaajalle ja käyttäjälle.
Kommentit kannattaa kirjoittaa aina!

Ohjelmakoodien perässä on kysymyksiä.
Kysymyksiin vastataan perässä olevaan Vastaus-kohtaan.
Komentojen perästä puuttuu kommentteja.
Kirjoita puuttuvat kommentit ja täydennä aloitetut.

Teemme alkuvalmistelut.
Tuomme ohjelmaan ohjelmaympäristöstä valmiina löytyvät kirjastot ja moduulit.
Nämä komentorivit täytyy kirjoittaa jokaiseen erilliseen ohjelmaan tällä kerralla.

```
import turtle                 # Ohjelmaan tuodaan turtle-moduuli,
                              # joka sisältää tarvittavat turtle-komennot.
kilpinen = turtle.Turtle()    # Annetaan kilpikonnalle nimi kilpinen
                              # eli luodaan kilpinen-niminen turtle-muuttuja.
kilpinen.shape("turtle")      # Tehdään kilpisestä kilpikonnan muotoinen.
```

```
# LÄMMITTELYTEHTÄVÄ 1 Neliö
# Piirretään neliö, jonka lähtöpiste on origo (0,0).
# Neliön sivun pituus on 100 pikseliä.
# Kilpisen etenemissuunta on alussa oikealle.

kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)          # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on ylöspäin.
kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)          # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on vasemmalle.
kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)          # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on _____
kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)          # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on _____

# KYSYMYS 1: Mitä koodille pitää tehdä, jotta kilpisen suunta lopussa on oikealle?
# VASTAUS 1: Lisätään nyt näkyvä viimeinen komentorivi: kilpinen.left(90) . Tämä puuttuu oppilaalta.

# KYSYMYS 2: Mitä koodille pitäisi tehdä, jotta neliö piirtyisi origosta vasemmalle ja origon alapuolelle?
# VASTAUS 2: Kopioi muokkaamasi koodi alle. Koodi on alla:

kilpinen.left(180)         #
kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä ja piirtää viivaa.
kilpinen.left(90)          # kilpinen kääntyy vasempaan 90 astetta.
kilpinen.forward(100)      # kilpinen liikkuu 100 pikseliä ja piirtää viivaa.
kilpinen.left(90)          #
kilpinen.forward(100)      #
kilpinen.left(90)          #
kilpinen.forward(100)      #
kilpinen.left(90)          #
```

```
# KYSYMYS 3: Mitä koodille pitäisi tehdä, jotta neliö piirtyisi origosta oikealle
# ja ensimmäisen neliön alapuolelle?
# Kilpinen pitää palauttaa origoon.
# VASTAUS 3: Kopioi koodi alle. Koodi on alla:

kilpinen.left(180)          # kilpinen käännetään 180 astetta.
kilpinen.forward(100)       # kilpinen liikkuu 100 pikseliä ja piirtää viivaa.
kilpinen.right(90)          # kilpinen kääntyy OIKEAAn 90 astetta.
kilpinen.forward(100)       # kilpinen liikkuu 100 pikseliä ja piirtää viivaa.
kilpinen.right(90)          #
kilpinen.forward(100)       #
kilpinen.right(90)          #
kilpinen.forward(100)       #
kilpinen.right(90)          #

# KYSYMYS 4: Mitä tapahtuu, kun suoritat seuraavan koodin?
kilpinen.clear()
# VASTAUS 4: Kaikki nollautuu ja kilpinen palaa origoon. Sen suunta on oikealle.

# KYSYMYS 5: Miten saat kaiken takaisin?
# VASTAUS 5: Useita tapoja. Ctrl + z, poista em. koodi, olethan tallentanut koodisi aiemmin
```

TEHTÄVÄ 2

```
# TEHTÄVÄ 2
# KOPIOI TÄSTÄ ALKAVA TEKSTI SUORAAN TRINKETIN OHJELMOINTI-IKKUNAAN!
# Tee tehtävä 2 eli Neliö käyttäen toistoa.

# Alkuun on taas liitetty alkutoimenpiteet.
# Ne pitää olla jokaisessa ohjelmassa.

# Alla on kertauksena neliön piirtävä ohjelma.

import turtle                 # Ohjelmaan tuodaan turtle-moduuli,
                              # joka sisältää tarvittavat turtle-komennot.
kilpinen = turtle.Turtle()    # Annetaan kilpikonnalle nimi kilpinen
                              # eli luodaan kilpinen-niminen turtle-muuttuja.
kilpinen.shape("turtle")      # Tehdään kilpisestä kilpikonnan muotoinen.

# LÄMMITTELYTEHTÄVÄ 1 Neliö
# Piirretään neliö, jonka lähtöpiste on origo (0,0).
# Neliön sivun pituus on 100 pikseliä.
# Kilpisen etenemissuunta on alussa oikealle.

kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)         # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on ylöspäin.
kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)         # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on vasemmalle.
kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)         # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on _____
kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
kilpinen.left(90)         # kilpinen kääntyy vasempaan 90 astetta. Sen suunta on _____
```

```
# KYSYMYS 1: Mitä koodille pitää tehdä, jotta kilpisen suunta lopussa on oikealle?
# VASTAUS 1: Lisätään nyt näkyvä viimeinen komentorivi: kilpinen.left(90) .

# Siirrämme ensin kilpisen alemmas.
# Emme halua, että piirrosjälki näkyy.
# Nostamme kynän ylös ja laskemme sen takaisin alas.

kilpinen.penup()            # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(0,-100)       # kilpinen siirtyy 100 pikseliä alaspäin.
kilpinen.pendown()          # kilpinen laskee kynänsä maan pinnalle.

# Teemme toistorakenteen eli silmukan.
# Nyt kilpinen kääntyy toiseen suuntaan.

for i in range(0,4):        #
  kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(90)        # kilpinen kääntyy oikeaan 90 astetta. Sen suunta on ylöspäin.

# Suorita ohjelmat.
```

TEHTÄVÄT 3-6

#TEHTÄVÄT 3-6

KOPIOI TÄSTÄ ALKAVA TEKSTI SUORAAN TRINKETIN OHJELMOINTI-IKKUNAAN!

Tee tehtävät 3 - 6 käyttäen toistoa.

Alkuun on taas liitetty alkutoimenpiteet.

Ne pitää olla jokaisessa ohjelmassa.

Alla on kertauksena neliön toistolla piirtävä ohjelma.

```
import turtle                 # Ohjelmaan tuodaan turtle-moduuli,
                              # joka sisältää tarvittavat turtle-komennot.
kilpinen = turtle.Turtle()    # Annetaan kilpikonnalle nimi kilpinen
                              # eli luodaan kilpinen-niminen turtle-muuttuja.
kilpinen.shape("turtle")      # Tehdään kilpisestä kilpikonnan muotoinen.
```

Siirrämme ensin kilpisen takaisin origoon.

Emme halua, että piirrosjälki näkyy.

Nostamme kynän ylös ja laskemme sen takaisin alas.

```
kilpinen.penup()              # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(0,0)            # kilpinen menee keskipisteeseen.
kilpinen.pendown()            # kilpinen laskee kynänsä maan pinnalle.
```

```
#Tehtävä 2 Neliö toistolla

# Teemme toistorakenteen eli silmukan.
# Nyt kilpinen kääntyy toiseen suuntaan.

for i in range(0,4):     # Opettaja selittää!
  kilpinen.forward(100)   # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(90)     # kilpinen kääntyy oikealle 90 astetta. Sen suunta on ylöspäin.

# Suorita ohjelma.

# TEHTÄVÄ 3 TASASIVUINEN KOLMIO

# Siirrä kilpinen toiseen paikkaan
# ja piirrä toistorakennetta käyttäen tasasivuinen kolmio.

# TÄRKEÄÄ: Kääntymiskulman suuruus ei ole sama asia
# kuin kolmion kulma (60 astetta).

# Kokeile ensin alla olevaa koodia ja muokkaa sitä niin, että saat kolmion.

kilpinen.penup()              # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(0,100)          # kilpinen menee 100 pikseliä alaspäin.
kilpinen.pendown()            # kilpinen laskee kynänsä maan pinnalle.


for i in range(0,3):          # Toista kolmesti. Kolmiossa on kolme sivua.
  kilpinen.forward(100)       # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(120)         # kilpinen kääntyy oikeaan 60 astetta. Sen suunta on ylöspäin.
```

```
# TEHTÄVÄ 4
# Käännä kolmio niin, että kolmion kärki osoittaa ylöspäin.

kilpinen.penup()            # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(0,100)        # kilpinen menee 100 pikseliä alaspäin.
kilpinen.pendown()          # kilpinen laskee kynänsä maan pinnalle.


for i in range(0,3):        # Toista kolmesti. Kolmiossa on kolme sivua.
  kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.left(120)        # kilpinen kääntyy oikeaan 120 astetta. Sen suunta on ylöspäin.

# TEHTÄVÄ 5 VIISIKULMIO
# Siirrä taas kilpinen toiseen paikkaan ja piirrä säännöllinen viisikulmio.

kilpinen.penup()            # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(0,-100)       # kilpinen menee 100 pikseliä alaspäin.
kilpinen.pendown()          # kilpinen laskee kynänsä maan pinnalle.


for i in range(0,5):        # Toista viidesti. Kolmiossa on kolme sivua.
  kilpinen.forward(100)     # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.left(360/5)      # kilpinen kääntyy oikeaan 72 astetta. Sen suunta on ylöspäin.
```

```
# TEHTÄVÄ 6 KUUSIKULMIO

for i in range(0,6):            # Toista viidesti. Kolmiossa on kolme sivua.
  kilpinen.forward(50)          # kilpinen liikkuu 100 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(360/6)

# Siirretään kilpistä hieman oikealle:

kilpinen.penup()                # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(50,-100)          # kilpinen siirtyy.
kilpinen.pendown()              # kilpinen laskee kynänsä maan pinnalle.

# Tehdään toinen kuusikulmio.

for i in range(0,6):            # Toista viidesti. Kolmiossa on kolme sivua.
  kilpinen.forward(50)          # kilpinen liikkuu 50 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(360/6)

# Nostetaan vielä kilpinen ylös tontun hattuun.

kilpinen.penup()                # kilpinen nostaa kynänsä irti maasta.
kilpinen.goto(50, 150)          # kilpinen siirtyy.
kilpinen.pendown()              # kilpinen laskee kynänsä maan pinnalle.

# KIVINEN TONTTU ON VALMIS!
```

#TEHTÄVÄT 7 ja 8

KOPIOI TÄSTÄ ALKAVA TEKSTI SUORAAN TRINKETIN OHJELMOINTI-IKKUNAAN!

Tee tehtävät 7 ja 8 käyttäen toistorakennetta.

Alkuun on taas liitetty alkutoimenpiteet.

Ne pitää olla jokaisessa ohjelmassa.

Alla on kertauksena kuusikulmion toistolla piirtävä ohjelma.

```
import turtle                 # Ohjelmaan tuodaan turtle-moduuli,
                              # joka sisältää tarvittavat turtle-komennot.
kilpinen = turtle.Turtle()    # Annetaan kilpikonnalle nimi kilpinen
                              # eli luodaan kilpinen-niminen turtle-muuttuja.
kilpinen.shape("turtle")      # Tehdään kilpisestä kilpikonnan muotoinen.
```

"Kopioidaan taas edellisen kuusikulmion koodi seuraavaan tehtävään.

```
for i in range(0,6):          # Toista viidesti. Kolmiossa on kolme sivua.
  kilpinen.forward(50)        # kilpinen liikkuu 50 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(360/6)
```

TEHTÄVÄ 7 PIIRRÄ OMIA MONIKULMIOITA, joissa on enemmän kulmia, kuin kuusikulmiossa.

Piirrä esimerkiksi 10-kulmio, 20-kulmio jne.

```
for i in range(0,10):         # Toista.
  kilpinen.forward(50)        # kilpinen liikkuu 50 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(360/10)      # Lasketaan kulman suuruus.

for i in range(0,20):         # Toista.
  kilpinen.forward(50)        # kilpinen liikkuu 50 pikseliä eteenpäin ja piirtää viivaa.
  kilpinen.right(360/20)      # Lasketaan kulman suuruus.
```

TEHTÄVÄ 8 PIIRRÄ JOKIN HIENO KUVIO; LOGO TAI VASTAAVA.

Avaa kokonaan uusi ohjelma Trinketissä.

Kopioi alkutoimenpiteet ja valmista koodia edellisestä tehtävästä.

KOODAA

NÄYTÄ OPETTAJALLE, MITÄ SAIT AIKAAN

PALAUTA TAAS CLASSROOMIIN.

LÄHTEET:

[1] Tie koodariksi -sivusto

https://tie.koodariksi.fi/alkeet/turtle

[2] Peilaus pisteen ja suoran suhteen Pythonin Turtle moduulilla - Oppilaan ohje

https://www2.helsinki.fi/sites/default/files/atoms/files/oppilaan_ohje_python_peilit.pdf

[3] Trinket.io -ohjelmointiympäristö

https://trinket.io/

Geometrian kuvioita ohjelmoimalla, 7.lk

Riitta Kotilainen, CC BY-SA 4.0

4.4.2021

Tuntisuunnitelma 7. luokalle (geometria)

Aihepiirin valinta ja rajaus

Ohjelmointia sovelletaan 7. luokan matematiikan opiskeluun geometrian osuudessa. On luontevaa ottaa aiheet matematiikassa käsiteltävistä aiheista, jolloin molemmat tukevat toisiaan. Tavoitteena on hahmottaa geometrian kuvioita ja hyödyntää ohjelmointia niiden piirtämisessä. Aihepiirissä käsitellään kolmion kulmien summaa, vieruskulmia, täyden kulman suuruutta (tämä lopuissa kortin tehtävistä). Vahvistetaan käsitteiden ymmärtämistä sovellustehtävissä. Oppilas tekee itsearvioinnin ja palauttaa koodauskortin opettajalle. Opettaja arvioi koko työskentelyn ajan oppilaita havainnoiden. Opettaja saa tietoa oppilaiden kyvyistä myös kohdasta: Kuka auttoi. Sekä ohjelmointia että kohdeoppimista arvioidaan kokonaisuutena. Koodauskortti jatkuu, mutta tässä tuntisuunnitelmassa keskitytään kahden tunnin osuuteen. Vain edistyneemmät saavat suunnikkaan ja tasakylkisen kolmion tehtyä.

Työskentelyvälineet ja opetusmenetelmät

Työskentely tapahtuu tietokoneilla. Käsitteitä on jo käyty läpi ennakkoon. Tätä ennen on tehty myös koodaustunnin harjoitus Frozen (https://studio.code.org/s/frozen/stage/1/puzzle/1). Opettaja ohjaa aluksi, kuinka ajatellaan ”nokan” liikkumista kääntymisissä ja kynän käyttöä. Oppilaat voivat neuvotella keskenään, mutta kukin tekee oman työnsä, vaikka hitaammin. Silloin oma onnistuminen motivoi. Mikäli oppilas hallitsee sekä matematiikan, että ohjelmoinnin, hän voi jatkaa koodauskortin tekemistä itsenäisesti. Opettaja kuitenkin seuraa, että oppilas käyttää mielekkäitä ohjelmoinnillisia ratkaisuja.

Alussa sana KOODAUSKORTTI (siis isolla kirjoitettuna) on esitetty binäärimuodossa, jonka avulla herätetään myös keskustelua. Muunnoksen voi tehdä osoitteessa: https://nickciske.com/tools/binary.php

01001011011011101111011011100100011000010111010101110011011010110111011100100111010001110100011010010000110100001010

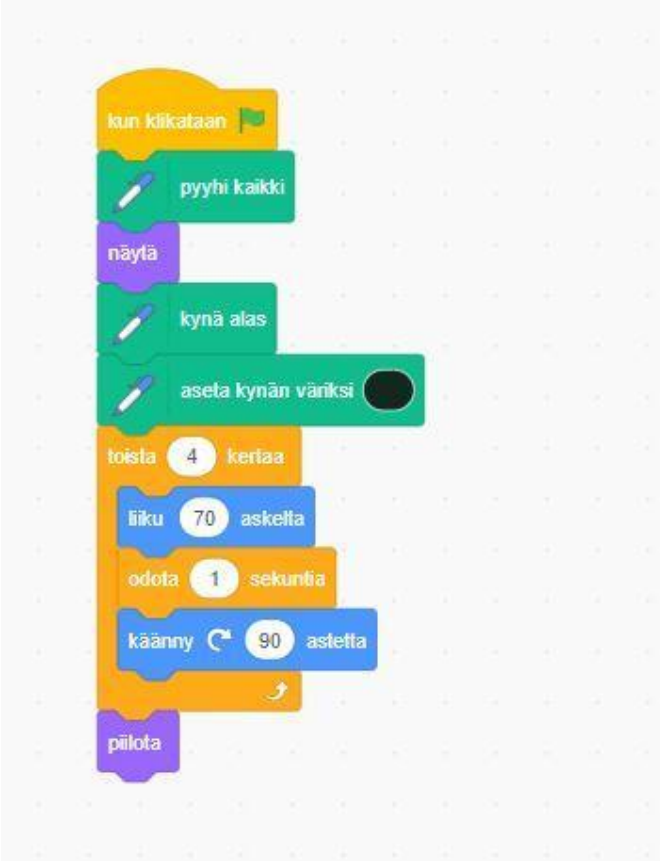
KOODAUSKORTTI Nimi: _____ Luokka: _____

Seuraavalla rivillä on kerrottu, mitä hymiöt tarkoittavat. Käytä niitä itsearvioinnissasi.

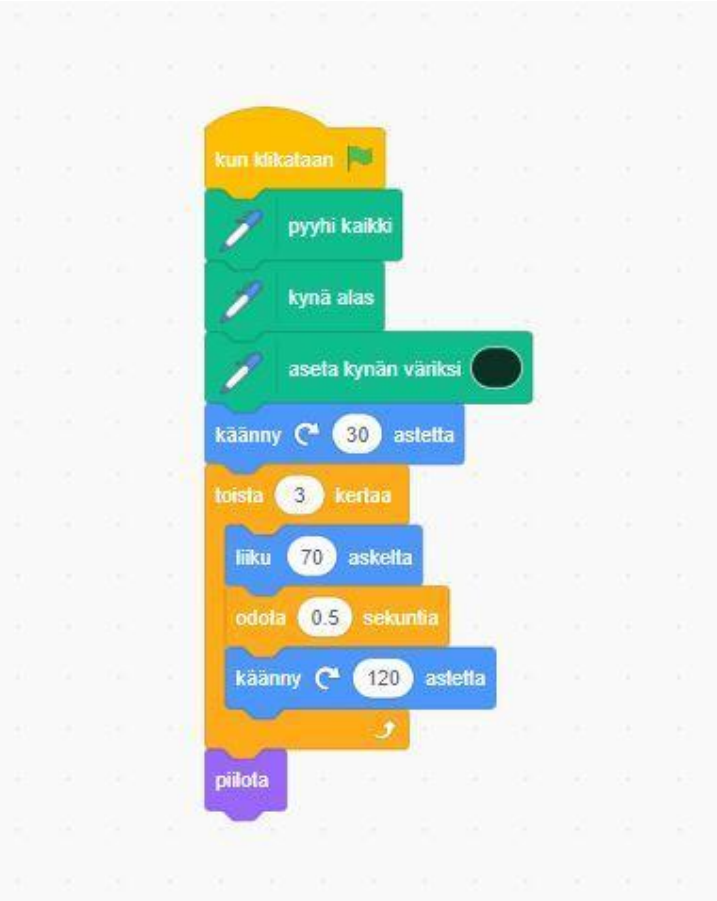
Helppoa! En tarvinnut apua. Rasti peukku, jos autoit myös muita.	Tarvitsin hieman apua opettajalta tai kavereilta.	Jouduin miettimään tosi paljon. Tarvitsin aika useassa kohdassa apua opettajalta tai kavereilta.	Tehtävä tuntui vaikealta. Tarvitsin paljon apua, mutta sain tehtyä!
Tehtävä	Ratkaisun hahmottaminen	Itsearviointi	Kuka auttoi?
Piirrä neliö, jonka sivu on 70. Tallenna nimellä neliö.			
Piirrä tasasivuinen kolmio, jonka sivun pituus on 70. Tallenna nimellä tasasivuinen kolmio.			
Tee suunnikas, jonka kulmat ovat 60 ° ja 120 °. Sivut 50 ja 100. Tallenna nimellä suunnikas. 			
Piirrä tasakylkinen kolmio, jonka huippukulma on 30° ja sivujen pituudet 90. Kolmion tulee olla tässä asennossa:  Tallenna nimellä tasakylkinen kolmio.			

Ratkaisut

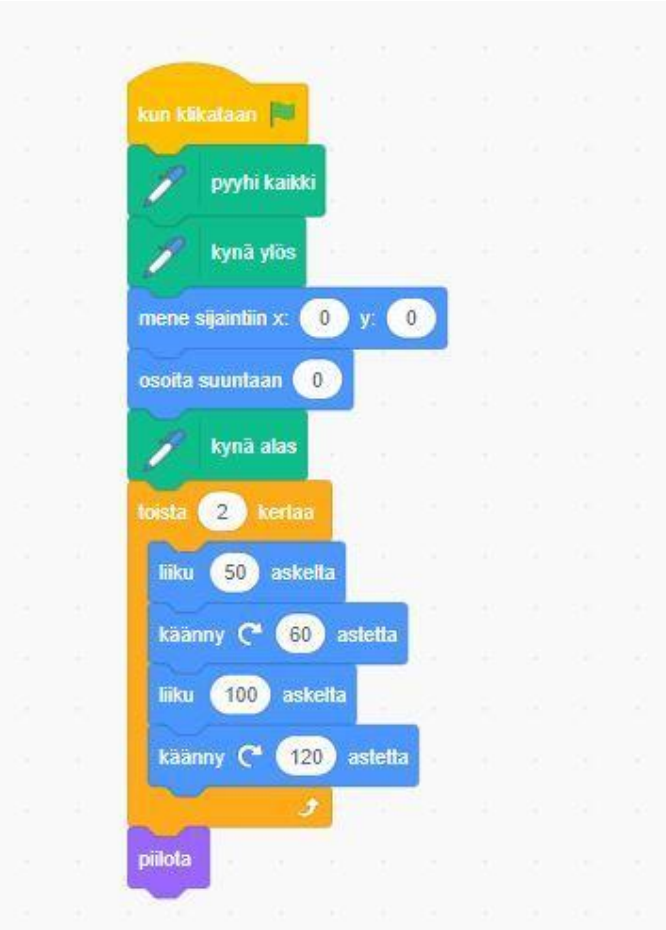
Neliö



Tasasivuinen kolmio



Suunnikas



Koordinaatiston kertaamista ohjelmoinnin avulla, 7. lk

22.2.2021

Tuntisuunnitelma 7 luokalle - ohjelmointia ja koordinaatistoa

Ohjelmointi ja algoritminen ajattelu ovat osa matematiikan opetussuunnitelmaa[1]. Halusin kuitenkin yhdistää ohjelmoinnin harjoitteluun myös toisen matematiikan sisällön eli koordinaatiston. Koordinaatiston käsite on 7 luokkalaisille jo tottu alakoulusta, mutta vaatii aina kuitenkin kertaamista. Ohjelmointi tarjoaa mielestäni motivoivan tavan kerrata koordinaatistoa.

Koordinaatiston opetteluun liittyvät oleellisesti akselit ja niiden nimeäminen sekä pisteiden koordinaattien tunnistaminen. Tällä tunnilla mielestäni on oleellista kerrata koordinaatiston pääpiirteet - mitkä ovat x- ja y-akseli ja miten hahmon liikkuminen koordinaatistossa vaikuttaa hahmon koordinaatteihin. Tunnin tavoitteet ovatkin: harjoitella ohjelmoinnin peruskäsitteitä ja ideaa scratchin avulla sekä kerrata koordinaatiston käsitettä ja pisteide nimeämistä koordinaatistossa.

Arviointi on lähinnä formatiivista ja se tapahtuu kokonaisuutena opettajan ohjaavan palautteen kanssa. Opettaja kiertää luokassa ja ohjaa oppilaita kysymyksin sekä antaa positiivista palautetta onnistumisista.

Ohjaavia kysymyksiä voisi olla muun muassa:

-Millainen toiminto tähän tarvitaan?

-Miten haluat, että hahmo reagoi?

-Lue ohjeet ääneen minulle. Mihin kohtaan tämä kannattaa laittaa/ mikä kannattaa muuttaa?

Oppilaat arvioivat omaa osaamistansa tunnin lopussa lyhyellä itse arviolla. Arviointi kysymyksiä ovat :

1. Kuinka huolella luit työohjeet?
2. Kuinka huolella työskentelit?
3. Saitko ohjelman toimimaan kuten halusit?
4. Kehittikö ohjelmaan jotain lisää mitä ohjeessa ei sanottu?
5. Autoitko kaveria/ teit yhteistyötä?

Koulumme 7 luokkalaisilla on käytössä chromebookit, joten käytämme chromeja työvälineinä. Tämä onnistuu hyvin sillä scratch toimii hyvin selaimella. Lisäksi jaan oppilaille ohje monisteet[2]. Ennen ohjeiden jakoa kertaan oppilaiden kanssa koordinaatiston, peruspiirteet (n 5 min) . Näin toivon, että he kiinnittävät scratchin lumoissakin huomiota helpommin koordinaatistoon. Käymme myös yhdessä läpi, miten scratchiin kirjaudutaan. Kerron ohjeistuksen oppilaille lyhyesti, jonka jälkeen he pääsevät lukemaan ohjetta.

Uskon, että oppilaita motivoi kun kerron, että tänään suunnittelemme oma pelin. Lisäksi tietokoneella työskentely motivoi monia oppilaita. Oppilaiden yhteistyötä rohkaistaan alkuunsa pyytämällä oppilaita auttamaan kaveria ja antamaan kaverille vinkkejä miten peliä

voidaan saada toimivammaksi. Lisäksi oppilaita voi kehottaa menemään kaverin viereen työskentelemään, jos luokka on sellainen, että kaverin vieressä pystytään työskentelemään.

Oppilaan aikaisempaa harrastuneisuutta voisi hyödyntää työssä hyvin eriyttämällä työtä. Työohjeessa oli jo valmiiksi hyviä ylöspäin eriyttäviä ideoita ja oppilaalta voisi kysyä myös itseltään, mitä hän haluaisi lisätä peliin, että siitä tulisi jännempi[3]. Uskon, että monilla oppilailla on kokemusta monista erilaisista peleistä ja näin ollen oppilaalla voi olla hyvin selkeä näkemys siitä, mikä tekee pelistä “ jännemmän”.

Uskon, että oppimis kokonaisuuden jälkeen oppilailla jää selkeämpi käsitys scratchista ja näin ollen tulevaisuudessa scratchiä on helpompi käyttää myös muiden tehtävien yhteydessä. Itse toivoisin, että voisin teettää luokkani kanssa tämän kyseisen oppitunnin jälkeen pienen pelin tai animaation suunnittelu projektin, jossa scratchin tuntemus olisi tarpeen.

1. https://www.oph.fi/sites/default/files/documents/perusopetuksen_opetussuunnitelman_perusteet_2014.pdf
2. https://www.cs.helsinki.fi/group/linkki/materiaali/lapsiohjeet/kerailypeli/kerailypeli.pdf
3. https://linkki.cs.helsinki.fi/cgi-bin/debbie-action-materiaalit?syn

Ohjelmointia Pythonilla, 7. lk

Svetlana Kallonen, CC BY-SA 4.0

11.10.2021

Tuntisuunnitelma Ohjelmointia Pythonilla 7. luokalle

7. luokan matematiikan tunnilla aloitetaan ohjelmoimaan Pythonilla.

(Edellisellä kerralla puhutaan algoritmisesta ajattelusta matematiikassa.)

Kahden tunnin aikana tavoitteena on tutustua lyhyesti ohjelmoinnin historiaan, kirjoittaa ensimmäinen ohjelma, oppia tulostamaan käyttämällä print komentoa, tutustua muuttuja käsitteeseen.

Ensin kysyn oppilailta mitä he tietävät ohjelmoinnista ja minkälaisia odotuksia ohjelmoinnista heillä on. Seuraavaksi puhumme algoritmisesta ajattelusta matematiikassa ja ohjelmoinnissa.

Sitten kerron oppilaille tavoitteista ja siitä, että oppilaat tekevät näiden pohjalla itsearvioinnin tunnin lopussa.

Tunneilla käytetään Tie-koodariksi sivuston kappaleiden 2 ja 3 tehtäviä.

Oppilaat tekevät itselleen tunnukset.

Ennen kun siirrytään tehtäviin, kerron print-komennosta ja muuttujista. Käydään Tie koodariksi esimerkkejä läpi yhdessä. Tunneilla tehdään kolmen ensimmäisen kappaleen tehtäviä. Oppilaat voivat valita tekevätkö itsenäisesti tai parin kanssa. Opettaja käy katsomassa miten menee, kehuu ja auttaa tarvittaessa.

Toisella tunnilla jätetään aikaa, jotta oppilaat ehtisivät keksiä oman tehtävän ja kirjoittaa sen ratkaisun.

Opettaja pyytää oppilaita esittämään oman tehtävän muille.

Sen jälkeen tekevät itsearvioinnin. Kun se on valmis, käydään läpi mitä on opittu.

Oppilaan Itsearviointi:

1. Tunnilla olen tutustunut ohjelmoinnin historiaan. Tiedän esimerkiksi millä vuosiluvulla ensimmäinen tietokone on keksitty. kyllä/ ei
2. Tiedän, miksi käytetään print komentoa. kyllä/ei
3. Osaan luoda muuttuja ja antaa sille arvo. kyllä/ on vielä epäselvä
4. Olen ratkaissut ainakin 4 tehtävää/ kappale. kyllä/ei

Lähteet:

Lumatikka Algoritmisen ajattelun kehittäminen

Tie koodariksi, kappaleet 1-3

Kilpikonnagrafiikkaa Pythonilla, 7. lk

Tiina Kauvosaari, CC BY-SA 4.0

19.3.2022

Kilpikonnagrafiikkaa Pythonilla 7lk

Aiheena 7 – luokan matematiikka ja geometria. Geometria on 7 – luokkalaisille hyvinkin tuttua alakoulusta. Haluan laajentaa heidän osaamistaan geometriassa ohjelmoinnin avulla. Valitsin tähän harjoitukseen erilaiset kolmiot.

Oppimistavoitteiden määrittely

Matematiikan oppimistavoite on osata nimetä, määritellä sekä piirtää erilaisia kolmioita. Kolmiot voidaan nimetä niiden ominaisuuksien mukaan sivujen pituuksien tai kulmien suuruuksien mukaan. Kolmiota on tasakylkinen, tasasivuinen, erisivuinen, suorakulmainen, tylppäkulmainen ja teräväkulmainen kolmio. Ohjelmoinnin oppimistavoite on osata piirtää erilaisia kolmiota. Lisäksi ohjelmointiosuudessa oppilaiden algoritminen ajattelu kehittyy, kun heidän täytyy suunnitella, miten erilaiset kolmiot voidaan ohjelmoimalla piirtää. Bonustehtävänä on käyttää ohjelmoinnissa värejä.

Arviointi

Tämä työ tehdään 2-3 hlön ryhmissä. Oppilaiden pitää suunnitella ja toteuttaa työ pythonilla. Työstä tehdään docs – tiedosto, johon oppilaat lisäävät koodit sekä luvankaappaukset töistä.

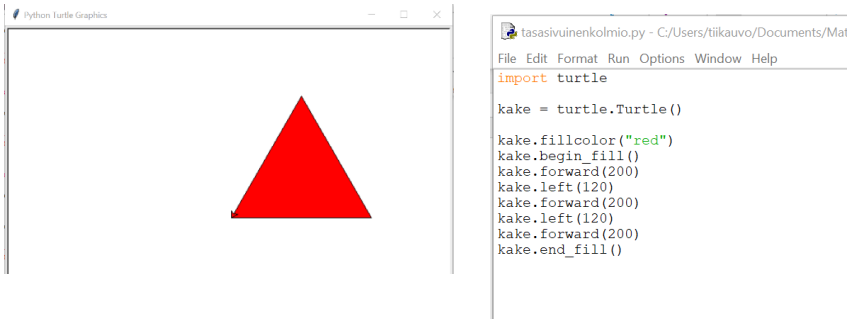
Opettaja arvioi työt. Oppilaat tekevät myös itse- ja vertaisarvioinnin, jossa pääpaino on työskentelyn arvioinnissa. Työssä arvioidaan sekä matematiikkaa että ohjelmointia ja siitä annetaan arvosana.

Työskentelyvälineet ja opetusmenetelmät

Opettaja näyttää aluksi kommennot joilla piirtäminen pythonilla tapahtuu. Erilaiset komennot ovat myös oppilaiden saatavilla tehtävänannossa. Oppilaille python-ohjelmointi ja editorin käyttö on jo ennestään tuttua, joten opettajan täytyy vain näyttää miten kilpikonnaohjelmointi tapahtuu.

Oppilaat saavat työn ohjeet kirjallisesti.

Tässä esimerkkikuva työstä.



Ehtorakenteiden harjoittelua Pythonilla, 7.-8. lk

14.10.2022

Python-ohjelmointi, valintarakenne if-else

Tämän harjoituksen tarkoituksena on oppia valintarakenteen if-else käyttäminen. Harjoitus toteutetaan tietokoneilla käyttäen Python-ohjelmointikieltä ja sitä Python-editoria, jota oppilaiden kanssa on aikaisemminkin käytetty, esimerkiksi https://replit.com/languages/python3 tai https://trinket.io/python.

Harjoitus toteutetaan matematiikan oppitunnilla parityöskentelynä sen jälkeen, kun oppilaiden kanssa on käyty läpi sekä neliön että suorakulmion piirin ja pinta-alan laskeminen. Harjoituksen on tarkoitus vahvistaa aikaisemmin opittuja asioita. Itse voisin kuvitella tekeväni tämän harjoituksen joko 7.luokan tai 8.luokan oppilaiden kanssa.

Oppitunnin aikana on tarkoitus luoda ohjelma, joka laskee annettujen tietojen avulla joko neliön tai suorakulmion piirin ja pinta-alan sekä tulostaa ne. Aikaa harjoitukseen on tarkoitus käyttää yksi 45 minuutin oppitunti. Mikäli edellisestä ohjelmointioppitunnista on paljon aikaa, on syytä käyttää tähän harjoitukseen aikaa kaksi oppituntia. Tällöin ensimmäisellä oppitunnilla keskitytään kertaukseen (vähintään 30 min). Ensimmäisen oppitunnin lopusta varataan aikaa noin 15 minuuttia valintarakenteen if-else esittelyyn sekä oman ohjelman suunnitteluun parin kanssa. Suunnitteluvaihe voidaan toteuttaa kynällä ja paperilla.

Jotta harjoituksen voi tehdä oppilaiden kanssa, oppilaiden tulee hallita seuraavat asiat Python-kielestä:

- print() -komento
- käsite muuttuja ja sen käyttö
- input() -komento
- tietotyyppi int() eli kokonaisluku
- vertailu

Tehtävä on siis tarkoitus toteuttaa parityöskentelynä, jolloin arvioinnin kohteina ovat sekä itse ohjelmakoodi että sen toimivuus ja parityöskentelytaidot. Oppilaita rohkaistaan olemaan aktiivisia, omia ajatuksiaan esille tuovia, pariaan kuuntelevia sekä palautetta antava ja vastaanottavia jäseniä. Arvioinnissa käytetään myös sekä vertaisarviointia että itsearviointia. Liitteessä 1 on harjoituksessa käytettävät arviointitaulukot. Parien muodostamisessa opettaja käyttää apunaan oppilastuntemusta. Mikäli ryhmässä on pariton määrä oppilaita, voi harjoituksen tehdä myös kolmen hengen ryhmässä.

Oppitunnin aikana oppilaiden on tarkoitus kirjoittaa koodi, jossa ohjelma ensin tervehtii käyttäjää, sen jälkeen pyytää käyttäjää tekemään valinnan neliön ja suorakulmion välillä. Tämän jälkeen ohjelma kysyy käyttäjältä joko neliön sivun pituuden tai suorakulmion kannan pituuden ja korkeuden. Annettujen tietojen avulla ohjelma laskee kuvion piirin ja pinta-alan sekä ilmoittaa ne käyttäjälle. Harjoituksessa on tarkoitus käyttää kokonaislukuja. Liitteessä 2 on malliratkaisu.

Mikäli ryhmässä on oppilaita, jotka ovat ohjelmoineet paljon, he voivat käyttää koodissaan desimaalilukuja (float() -komento) sekä harjoitella tuloksien pyöristämistä (round ()-komento). Tai oppilaat voivat harjoitella valintarakennetta if-elif-else. Rakenne antaa enemmän mahdollisuuksia valintaan. Oppilaat voivat kirjoittaa koodin, jossa valitaan kolmen (esim. neliö, suorakulmio ja kolmio) tai useamman eri kuvion väliltä (esim. neliö, suorakulmio, kolmio ja suunnikas).

Python kielessä float()-komento tarkoittaa desimaalilukua. Jos haluaa käyttää desimaalilukuja, on muistettava, että desimaalimerkkinä käytetään pistettä. Desimaalilukujen pyöristäminen haluttuun tarkkuuteen tapahtuu käyttämällä round()-komentoa.

Mikäli tehtävää käytetään aikaa yksi oppitunti, alkaa oppitunti lyhyellä kertauksella. Kertauksessa käydään esimerkkikoodin avulla aikaisemmin opiskellut asiat (komennot print() ja input(), muuttujat, tietotyyppi int() sekä laskujen kirjoittaminen). Samalla palautetaan mieleen, että koodin kommentointi auttaa sekä itseä muistamaan omia ajatuksia että muita koodin ymmärtämisessä.

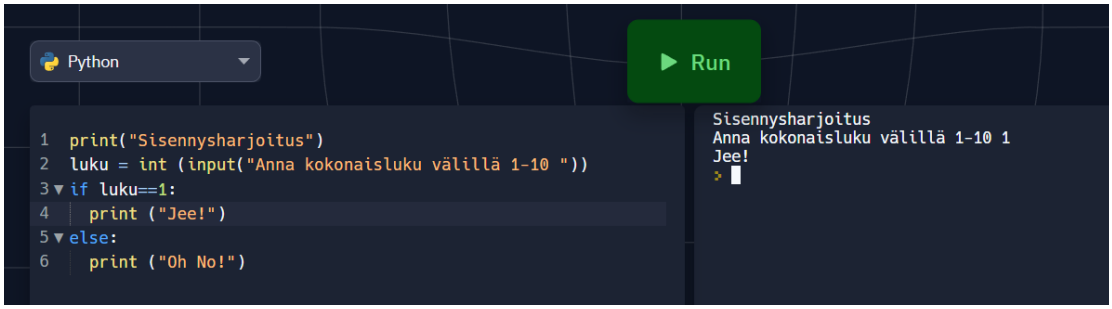
Ohessa esimerkkikoodi, jota voi käyttää kertauksen apuna. Koodi on kirjoitettu käyttäen https://replit.com/languages/python3. Vain kirjoitettu koodi näytetään oppilailla, heidän tehtävänään on kertoa mitä koodi tekee. Tämän jälkeen koodia kokeillaan yhdessä.

Mikäli harjoitukseen käyttää aikaa kaksi oppituntia, kannattaa oppilaiden antaa itse kirjoittaa kertaavia harjoituksia. Ohessa esimerkkejä kertaavista harjoituksista.

- Kirjoita koodi, joka tulostaa tervehdyksen.
- Kirjoita koodi, joka pyytää käyttäjää syöttämään kaksi kokonaislukua ja tämän jälkeen tulostaa lukujen summan.

Lyhyen kertauksen jälkeen käydään läpi if-else rakenne. Rakenteella voidaan tehdä valintoja erilaisilla ehdoilla. Samassa yhteydessä muistutetaan, että Python -kielessä on omat säännöt, joita tulee noudattaa koodia kirjoitettaessa. Esimerkiksi välilyönti tai sisennys rivin alussa on merkitsevä. Se määrittelee mihin kokonaisuuteen rivi loogisesti kuuluu. Ne osat koodia, jotka halutaan liittyvän toisiinsa, tulee asettaa samalla sisennystasolle. Koodirivejä ei siis voi aloittaa ihan mistä vaan.

Ohessa esimerkkikoodi, jonka avulla voi selittää oppilaille valintarakenteen if-else periaatetta sekä käydä läpi vertailu (rivi 3). Koodi on kirjoitettu käyttäen https://replit.com/languages/python3.



Oppilaille tulee myös kertoa, että if-rivin ja else-rivin loppuun tulee laittaa kaksoispiste. Muuten ohjelmaa suoritettaessa tulee virhekoodi.

Oppitunnin lopusta pitää varata muutama minuuttia aikaa siihen, että oppilaat palauttavat koodinsa opettajalle (esim. kuvakaappaus tekstinkäsittelyohjelmaan ja tiedoston palauttaminen Teams:iin). Oppilaiden tulee myös tehdä tunnin lopussa sekä vertais- että itsearvioinnin. Vaihtoehtoisesti vertais- ja itsearviointi voidaan antaa oppilaille kotitehtäviksi.

Liite1 **Arviointitaulukot**

Opettajan arviointitaulukko koodia varten

	täysin oikein	**melkein oikein**	**osittain oikein**	**vähän oikein**
print-komento				
muuttuja-komento				
input-komento				
if-else-rakenne				

Opettajan arviointitaulukko parityöskentelyä varten

	hyvin	**osittain**	**ei ollenkaan**
tasapuolinen työnjako			
tasapuolinen osallistuminen			
parin kanssa keskustelu			

Vertaisarviointitaulukko oppilaalle

	hyvin	**osittain**	**ei ollenkaan**
Parini osallistui aktiivisesti.			
Parini kertoi ajatuksiaan.			
Sain palautetta pariltani.			

Itsearviointitaulukko oppilaalle

	hyvin	**osittain**	**ei ollenkaan**
Kuuntelin pariani.			
Osallistuin aktiivisesti.			
Työnjako oli tasapuolinen.			
Annoin parilleni positiivista palautetta.			
Osaan käyttää print-komentoa.			
Osaan käyttää muuttujia.			
Osaan käyttää input-komentoa.			
Osaan käyttää if-else-rakennetta.			

Liite 2 **Malliratkaisu**

Malliratkaisu on tehty käyttäen https://replit.com/languages/python3

Geometrisia kuvioita Turtlen avulla, 7.-9. lk

Anna Norrkniivilä, CC BY-SA 4.0

13.3.2021

Tuntisuunnitelma ohjelmointiin: Turtle ja monikulmiot (2 h)

Tarkoitus on harjoitella piirtämään erilaisia geometrisia kuvioita matematiikan tunnilla. Ohjelmoinnin avulla oppilas joutuu miettimään kulmien suuruksia enemmän ja tarkemmin kuin käsin piirtämällä. Kuitenkin oppilas pääsee nauttimaan ohjelman tarkkuudesta ja kyvystä toistaa samaa käskyä. Tästä syystä keskitymme tunnilla säännöllisiin geometrisiin monikulmioihin ja ympyröihin. Lopuksi on tarkoitus antaa aikaa luovuudelle suunnittelemalla oma logo tai muu kuvio, jotta oppilas saa mukavan kokemuksen ohjelmoinnista ja siinä onnistumisesta.

Oppimistavoitteena tässä kokonaisuudessa on tutustua Turtle-kirjastoon ja sillä piirrettävään grafiikkaan. Lisäksi ohjelmoinnin osalta tarkoitus on käyttää for-silmukkaa säännöllisen kuvion piirtämiseen. Matematiikan osalta harjoitellaan kulmia sekä tutustutaan erilaisiin säännöllisin monikulmioihin ympyröiden lisäksi. Oletus on, että oppilaat ovat tutustuneet Python-ohjelmointiin peruskomentojen verran.

Arviointia varten tehtävän lopussa on pistetaulukko, johon oppilas arvioi omaa osaamistaan hymynaamojen avulla. Arviointi tehdään oppilaan palauttaman koodin perusteella. Pääasiassa arvioidaan ohjelmointiosaamista, mutta lopputulos ei näytä oikealta ilman ymmärrystä geometrisista kuvioista. Tässä kuvioita on tosin melko vähän, jotta tehtävät olisi mahdollista saada tehtyä kahdessa tunnissa.

Tunnilla käytetään läppäreitä sekä Säde-kirjasarjan selainpohjaista Python-editoria. Vaihtoehtoisesti voi käyttää koneelle asennettua Pythonia tai jotain toista selainpohjaista Python-editoria. Editoria valitessa täytyy huomioida mahdollisuus Turtle-kirjaston käyttöön. Tehtävät kopioidaan muistioon ja muistio tallennetaan yhdessä sovittuun palautuskansioon.

Opettajan kannattaa tehdä esimerkki yhdessä oppilaiden kanssa. Tämän jälkeen oppilaat aloittavat tehtävät, joita saavat tehdä omaan tahtiin. Oppilaita kannustetaan kysymään apua kaverilta tai opettajalta. Usein työ sujuu paremmin, kun oppilaat istuvat pareittain. Viimeisessä tehtävässä kaikki pääsevät näyttämään osaamistaan, jota voi toki olla kertynyt muualtakin kuin tästä tehtävästä. Lopussa on linkki, josta oppilaat voivat etsiä lisäkomentoja, jos annetut komennot eivät riitä. For-toistorakennetta voidaan käyttää muissakin ohjelmointitehtävissä kuin piirtämisessä.

Lähteet: Säde-oppikirjan Python harjoitus 3, Ari Heimosen Maolin koulutuksessa jakama ohjelmointiharjoitus 2 sekä
https://peda.net/jyvaskyla/ict/palvelut/ohjelmointi-robotiikka/pop/l6ko

Hymynaamat: https://yle.fi/uutiset/3-5392462 (Evira/oivahymy.fi)

Ohjelmointia kilpikonnan avulla

Tarvittavat komennot:

Aloita työsi kirjoittamalla seuraavat komennot. Voit nimetä kilpikonnan haluamallasi tavalla.

```
from turtle import*     # Tuodaan turtle-komennot sisältävä kirjasto.
                        # Muilla pohjilla voi toimia myös: import turtle
alue = Screen()         # Määritetään piirtoalue.
otus = Turtle() # Annetaan kilpikonnalle nimi otus.
otus.reset()                    # Tyhjentää piirtoalueen ja palauttaa kilpikonnan origoon
                                oletusasetuksilla.
```

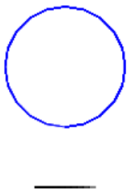
Muita komentoja:

```
otus.pendown()              # kynä alas
otus.penup()                # kynä ylös
otus.home()                 # palaa origoon
otus.goto(x,y)              # menee pisteeseen (x, y)
otus.forward(x)             # menee eteenpäin x askelta
otus.backward(x)            # menee taaksepäin x askelta
otus.left(α)                # kääntyy vasemmalle kulman α verran
otus.right(α)               # kääntyy oikealle kulman α verran
otus.pensize(x)             # asettaa viivan paksuudeksi x yksikköä
otus.color(”väri”)          # viivan väri muuttuu, väri englanniksi
otus.pencolor(”väri”)       # vaihtaa myös värin
alue.bgcolor(”väri”)        # asettaa taustan värin halutuksi
otus.fillcolor(”väri”)      # asettaa täyttövärin
otus.begin_fill()           # aloittaa täytön (juuri ennen, kun piirrät täytettävän kuvion)
otus.end_fill()             # lopettaa täytön (juuri piirretyn kuvion jälkeen)
otus.circle(x)              # piirtää ympyrän, jonka säde on x
otus.hideturtle()           # piilottaa kilpikonnan
otus.showturtle()           # tekee kilpikonnan näkyväksi
for i in range(n):          # toistorakenne, joka toistaa seuraavan komennon n kertaa
```

Esimerkki:

```
from turtle import*
alue = Screen()
otus = Turtle()
otus.reset()

otus.forward(50)        # piirretään 50 askelta pitkä viiva
otus.penup()            # nostetaan kynä ylös
otus.goto(25, 50)       # siirrytään viivan yläpuolelle
otus.pendown()          # kynä alas
otus.pencolor("blue") # vaihdetaan väri siniseksi
otus.circle(50) # piirretään ympyrä, jonka säde on 50 askelta
otus.hideturtle()       # piilotetaan kilpikonna
```

Lopputulos:



Tehtävät:

Luo muistio, jonka nimeät omanimi_kilpikonnaharjoitus. Tallenna tehtävien koodit tähän muistioon otsikoiden esim. Tehtävä 1 alle. Palauta muistio palautuskansioon ”Kilpikonnaharjoitus”.

1. Piirrä neliö, jonka sivun pituus on 50 askelta. Vaihda viivan väri punaiseksi.
2. Piirrä edellisen kohdan neliö käyttäen for-toistorakennetta.
3. Piirrä kaksi neliötä ja täytä ne eri väreillä.
4. Piirrä tasasivuinen kolmio, jonka sivun pituus on 150 askelta.
5. Piirrä säännöllinen kuusikulmio. Saat itse päättää sivun pituuden ja viivan värin.
6. Aseta uusi taustaväri. Piirrä ympyrä, jonka säde on 100.
7. Suunnittele logo tai kuvio, jolla pääset näyttämään osaamistasi.

Kysy tarvittaessa opelta tai kaverilta apua! Lisää komentoja löydät esimerkiksi osoitteesta http://www.turtlesrock.org/commands.html

Tehtävien arvostelu:

	Oma arvio 	Opettajan arvio
Tehtävä 1		/4 p
Tehtävä 2		/4 p
Tehtävä 3		/4 p
Tehtävä 4		/4 p
Tehtävä 5		/4 p
Tehtävä 6		/4 p
Tehtävä 7		/4 p
Yhteensä		/28 p

Python, 8.lk

Liliana Peuhkuri, CC BY-SA 4.0

5.5.2019

1. Aihepiirin valinta ja rajaus

Valitsin esimerkkisuunnitelma Python ohjelmoinnin integroimiseksi matematiikan opetukseen 8. vuosiluokalla. Aiheekseni on harjoitella kirjoittamaan ohjelmakoodia, joka suorittaa yksinkertaisia laskutoimituksia, piirtää geometrisia kuvioita ja lopuksi voidaan luoda funktio joka kuvaille matematiikka ja luonto esimerkki Fibonaccin lukusarjaa.

2. Oppimistavoitteet ja arviointi

Matematiikka ja ohjelmoinnin oppiminen on tärkeä ottaa se tehokäyttöön jo peruskoulussa, uuden opetussuunnitelman 2016 mukaan: *Ohjelmoinnin avulla oppii myös yleisesti hyödyllisiä kognitiivisia taitoja. Looginen ja luova ajattelu, tarkka työskentely, kyky hahmottaa ongelma ja muodostaa sille erilaisia ratkaisuvaihtoehtoja sekä visualisoida ja käsitteellistää ne ovat hyödyllisiä kaikissa aineissa ja elämässä itsessään*.

Tavoitteena on saada oppilas ymmärtämään Python-ohjelmointikielen kuten perus laskutoimituksia, peräkkäisrakenne, tulostus, merkkijono, lukujen vertailu, ehtorakenne ja geometristen kuvioiden piirtämien (Turtle-kirjaston funktioiden avulla). Oppimistavoitteena on myös, että oppilas voisi soveltaa erilaisissa oppiaineissa mahdollisimman paljon sekä oppilaan mielenkiinto ohjelmoinnin kohde.

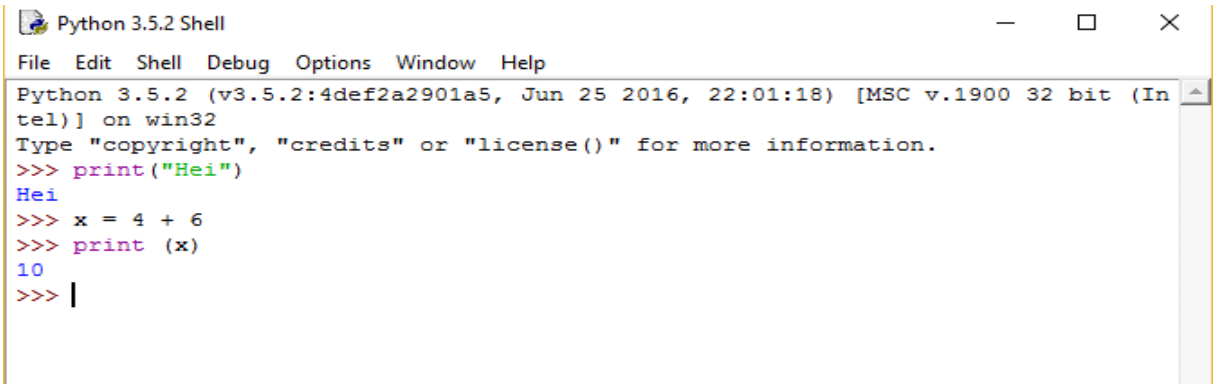
3. Työskentelyvälineet ja opetusmenetelmät

Tämä materiaali on kahden matematiikan tunti suunniteltu käytettäväksi Python-ohjelmointia joka toteutetaan matikan luokassa, jossa jokaisella oppilaille on oma kone käytettävissä.

Oppilaita kannustetaan sosiaaliseen opiskeluun oppilaslähtöisesti, eli heitä kehoittaan olemaan aktiivisia tunnilla ja ohjaamaan ja antamaan vinkkejä toinen toisilleen.

Tunnin alussa on kerrotettavaa miten ohjelmoidaan Pythonilla. Uuden ohjelman luominen ja tulkkaaminen ohjelmointiympäristössä:

- Helpoin tapa aloittaa Pythonin käyttö on ajaa Python-koodi IDLEllä. Ympäristö toimii kahdessa eri tilassa. Ensimmäinen tila (Shell) on komentorivitila, johon ympäristö ensin käynnistyy:



- Toisessa tilassa kirjoitetaan ohjelmia. Pääset tilaan luomalla uuden ohjelman: valitse Shell-tilassa File ja New File. Nyt voit aloittaa ohjelmasi kirjoittamisen:
- Kirjoita koodi.
- Tallenna tiedosto nimellä tehtävä.py.
- Ohjelman tallennuksen jälkeen voit ajaa ohjelmasi valitsemalla Run valikosta kohdan Run Module (tai paina F5):

Ensimmäinen oppitunnin suunnitelma(60 min).

Oppilaat täyttävät itsearviointilomakkeet kahdessa osassa: ennen tunnin aloittamista ja sen jälkeen kun oppitunnin on pidetty.

Täytä tämä alkuarviointi ohjelmoinninjakson aluksi.

Oma toimintani	Osaan hyvin	Osaan jotenkuten	En muista/ En ole varma	En osaa ollenkaan
Osaan opiskella asioita kirjasta/moniste.				
Osaan käyttää opetusvideoita opetuksen tukena.				
Osaan työskennellä yksin.				
Osaan työskennellä parin kanssa.				
Ohjelmoinnin taitoni				
Osaan ohjelmoida				
Osaan käyttää Python-ohjelma				
Osaan kirjoittaa yksinkertaisia käskyä / kommenti				
Osaan kirjoittaa peruslaskutoimitukset Pythonilla				
Osaan				

Tekstin tulostaminen tietokoneen ruudulle vaatii komennon. Tällainen komento Pythonissa on **print()**. Ohjelmaan voi kirjoittaa kommenttirivejä, joita ei tulkata eli ohjelma ei suorita niittä. Rivin alkuun on laitettava **#**-merkki.

Esimerkki:

```
print ("Opimme ohjelmoimaan Pythonilla.")
# tämä on kommenttirivi, jota ei tulkata.
```

Tehtävä 1. Tee ohjelma, joka tulostaa ja rivittää tekstin samoin kuin alla.

```
Pullaohje
===========
Maitoa:           5dl
Hiivaa:           25g
Suolaa:           1tl
sokeria           1,5dl-2,0dl
kanamunia         2kpl
Vehnäjauhoja:     14dl
Voita:            100g
Voiteluun kanamuna
Koristukseksi raesokeria
>>>
```

Muuttuja:

Esimerkki:

```
x = 5
y = 9
x = 5 # annetaan muuttujien x ja y arvoksi 5 ja 9.
y = 9
print (x+y) # lasketaan ja tulostetaan muuttujien summa.
print (x,y) # tulostetaan erikseen muuttujien arvo.
```

Tehtävä 2. Merkitään muuttujien arvoksi x=4, y=10 ja z=29. Laske ja tulosta ohjelman avulla.

Syöte:

Ohjelmissa tarvitaan usein syötteittä käyttäjiltä. Käyttäjältä voidaan pyytää syötettä komennolla **input():**

Esimerkki:

```
nimi = input("Anna nimesi:")  # kirjoita nimesi ja paina enter
ika = input("Anna ikäsi:") # paina enter
print ("Nimesi on",nimi,"ja ikäsi on",ika,"vuotta.")
```

Tehtävä 3.

Kirjoita ohjelma, joka pyytää käyttäjältä kaksi kokonaislukua, laskee niiden summan ja tulostaa tuloksen.

Tehtävä 4.

Kirjoita ohjelma, joka pyytää ympyrän säde, laske ympyrän pinta-ala sekä ympyrän kehän pituus.

vinkki tuloste:

```
Anna ympyrän säde: 4
Pinta-ala on 50.24
Kehä on 25.12
>>>
```

Lukujen vertailu

Lukuja ja vaikkapa merkkijonoja voi verrata keskenään käyttämällä vertailuoperaattoreilla. Vertailu tulos on joko tosi(True) tai epätosi(False).

Operaattori	Nimi	Esimerkki
<	Pienempi kuin	5 <3 palauttaa arvon False ja 3 < 5 palauttaa arvon True.
>	Suurempi kuin	5 >3 palauttaa arvon True.
<=	Vähemmän, tai yhtä suuri	Jos x = 3 ja y = 6 niin x <= y palauttaa arvon True.
>=	Suurempi, tai yhtä suuri	Jos x = 4 ja y = 3 niin x >= y palauttaa arvon True.
==	Yhtä suuri kuin	Jos x = 2 ja y = 2 niin x == y palauttaa arvon True. Jos x = 'str' ja y = 'stR' niin x == y palauttaa arvon False. Jos x = 'str' ja y = 'str' niin x == y palauttaa arvon True.
!=	Erisuuri kuin	Jos x = 2 ja y = 3 niin x != y palauttaa arvon True.

Valintarakenne if-else

if-else-rakenteella voidaan tehdä monivaiheisia valintoja eri ehdoilla. Jos ehdo on tosi (true), suoritetaan ehdon jälkeiseen kokonaisuuteen kuuluvat komennot. Koodirivejä ei saa aloittaa samoista paikoista, vaan on noudettava sisennyksien sääntöjä.

Esimerkki:

```
luku = int(input("Anna kokonaisluku välillä 1-10:"))
if luku > 5:
        print("Luku on suurempi kuin 5")
print("Tämä tulostetaan joka tapauksessa.")
```

Tehtävä 5.

Kirjoita ohjelma, joka kysyy käyttäjän iän ja tulostaa seuraavat tulosteet annetun iän mukaan.

Ikä	Tuloste
0…12	Lapsi
13…17	Nuori
18…22	Nuori aikuinen
23…	Aikuinen

Toinen oppitunnin suunnitelma(60 min).

Matematiikka ja luonto

LUONNON symmetria on kuin taikaa, ja siksi se on kiehtonut ihmisiä aina.

Voiko matematiikkaa nähdä luonnossa omin silmin? Kyllä, kun poimit päivänkakkaran ja katsot sen kierteistä siemenkotaa, näet muotoja ja kaavoja, joita ratkomalla tiedemiehet ovat ratkaisseet monta maailman mysteeriä.

Moni täysin tajuttomalta kuulostava matemaattinen teoria on myöhemmin pilkistänyt esille luonnossa, kun maailmankaikkeuden ilmiöitä on ymmärretty paremmin.

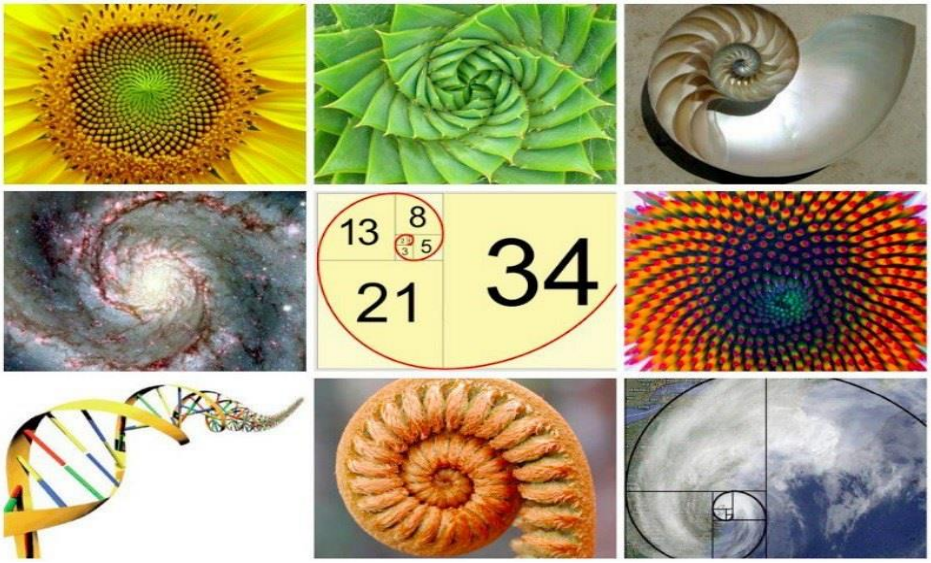
Kun nypit päivänkakkarasta terälehtiä ja mietit, rakastaako vai eikö rakasta, laskepa, montako terälehteä kukassa on. 13, 21 vai 34? Usein 34. Nämä luvut ovat osa Fibonaccin lukusarjaa.

Fibonaccin lukuja ovat 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,…

Lukujonon jäsenet muodostetaan kahden edellisen luvun summana.

0 + 1 = **1**, 1 + 1 = **2**, 1 + 2 = **3**, 2 + 3 = **5**, 3 + 5 = **8**,…

Monet luonnon tapahtumat ”toteuttavat Fibonaccin lukuja kasvaessaan”.



Kukilla on useimmiten Fibonaccin lukujen verran terälehtiä. Tällöin terälehdet mahtuvat mahdollisimman pieneen tilaan ilman, että ne menevät toistensa päälle. Se on niille edullisin tapa kasvaa.

Luonnossa esiintyy myös usein kultainen kulma 137,5°. Esimerkiksi lehdet asettuvat varren ympärille siten, että vierekkäisten lehtien välinen kulma on 137,5°.

Voimme luoda funktion, joka kirjoittaa Fibonaccin sarja mielivaltaiseen rajan:

Esimerkki:

```
a, b = 0, 1
while b < 1000:
      print(b),
      a, b = b, a+b
```

Täytä tämä ohjelmoinninjakson päätyttyä.

Matematiikan sisältöjen oppiminen	Hyvin	Melko hyvin	Jotenkuten	En lainkaan
Ymmärrän, mitä tarkoita muuttuja.				
Ymmärrän, mitä tarkoittaa komento print.				
Ymmärrän, mitä tarkoittaa syöte, tyyppimuunnoksia				
Osaan käyttää vertailuoperaattorit(True/False)				
Osaan muodostaa valintarakenne if-else				
Osaan kirjoittaa yksinkertaisia käskyä / kommentit				
Osaan kirjoittaa peruslaskutoimitukset Pythonilla				
Kokonaisuuden arviointi (Minkä arvosanan 4-10 antaisit itsellesi?)	Arvosana			

Kerro, miltä ohjelmoinnin opiskelu on tuntunut? Onko yläasteen matikka vastannut odotuksiasi?

Kuinka olet onnistunut omasta mielestäsi? Missä asioissa olet onnistunut?

Pohdi, mitkä asiat ovat olleet tunneilla OK, mitä muuttaisit opiskelussamme.

Mitä muuta haluaisit sanoa? Muuta palautetta.

LÄHTEET

http://koodi2016.fi/ops.html

Luvun muodostaminen (Scratch-ohjelmontia), 8. lk

24.1.2021

Luvun muodostaminen

Tunti on suunniteltumatematiikan tunnille 8. luokkalaisille, joille sekä Python- että Scratch-ohjelmointi ovat jo jonkin verran tuttuja. Tehtävä pohjautuu Antti Laaksosen Ohjelmointia matematiikkaan vihkoon https://mooc.helsinki.fi/pluginfile.php/152886/mod_resource/content/2/ohjelmointiaMatematiikanOpetukseen.pdf . Se löytyy vihosta ensimmäisenä ja on nimeltään Luvun muodostaminen. Tarkoitus on siis päästä mihin tahansa lukuun joko kertomalla luku kahdella tai lisäämällä lukuun 3 ja toistamalla tätä riittävän monta kertaa. Jotta annettu tehtävä konkretisoituu oppilaille, lähestymme ensin tehtävää Scrach-ohjelmalla, jonka avulla koodasin ohjelman, jolla voi etsiä reittiä pyydettyyn lukuun.

Ohjelma löytyy täältä https://scratch.mit.edu/projects/478116243/ Oppilaiden kanssa kokeillaan ohjelmaa ja mietitään onko mahdollista päästä kaikkiin lukuihin aloittamalla ykkösestä ja aina kertomalla kahdella tai lisäämällä lukuun kolme. Päädytään yhdessä tulokseen, että kolmella jaollisilla luvuilla algoritmi ei toimi. Isompia lukuja tutkittaessa huomataan, että yrityksen ja erehdyksen kautta toimittaessa aikaa voi kulua kauan. Samalla tulee kerratuksi peruslaskutoimmituksia ja kolmella jaollisuuden sääntö.

Sen jälkeen näytetään oppilaille valmis vihossa esitelty koodi ja mietitään yhdessä miten se toimii ja havaitaan, että koodi löytää vastauksen myös kolmella jaollisille luvuille ja mietitään miten se on mahdollista. Yhteisten pohdintojen jälkeen todetaan, että koodi toimii väärin, josta päästään hyvin keskusteluihin siitä, että koneisiin ei voi sokeasti luottaa.

Jos asiaan käytetään kaksi oppituntia oppilaat voivat katsoa mallia laskurin toteuttamiseen valmiista Scratch-projektista ja taiteelliset yksilöt voivat innostua tekemään laskurin painonapeista visuaalisesti hienoja.

Tehtävää voi helposti eriyttää alaspäin niin, että oppilas ei tee koodia ollenkaan itse vaan yrittää vain päästä laskutoimituksilla pyydettyyn lukuun. Ylöspäin voi eriyttää niin, että oppilas yrittää itse nähtyään mallin toteuttaa python-ohjelman, joka suoriutuu tehtävästä. Jos halutaan käyttää asiaan kaksi oppituntia oppilaat voivat itse Scratchillä ohjelmoida joko tämän saman laskurin tai sitten jonkin muun laskurin, joka toimii.

Esim. helpoimmillaan voisi tehdä laskurin, joka lisää aina klikattaessa laskuriin yhden. Tämä toiminto voisi olla kätevä esim. jonkin tapahtuman yhteydessä ovella, kun lasketaan kävijämäärää.

Arviointi tapahtuu oppilaita ja opetuskeskustelua havainnoimalla sekä, jos oppilaat pystyvät ohjelmoimaan joko Scrachillä tai Pythonilla tai molemmilla, valmiita ohjelmia tarkastelemalla. Työn voi tehdä myös pareittain tai pienissä ryhmissä, joka nyt korona-aikaan ei ole oikein suositeltavaa.

Python-ohjelmoinnin alkeet, 8. lk

13.4.2021

Valitsen Python ohjelmoinnin alkeisiin tutustumisen Tie koodariksi-sivuston tehtävien kautta. Käytän tähän 4 oppituntia 8. lk matematiikassa. Tehtävät arvosteltaisiin etenemisen pohjalta ts. muutaman tehtävän suoritus vastaisi tyydyttävää arvosanaa, mutta hyvälle ja kiitettävälle tasolle päästäkseen oppilaan tulisi tehdä 10-20 tehtävää.

Oppilaat saa motivoitua tehtävään sillä, että ahkeruudesta palkitaan ja toisaalta koodaus on jotain erilaista tekemistä, mitä yleensä matematiikan laskeminen on. Oppimistavoitteena on ohjelmointikoodin perusrakenteeseen tutustuminen ja eri rakenteiden nimeäminen valmiista koodista (esim. muuttujat, ehtolauseet, toiminnat, käskyt, looppi-rakenteet). Oppilas oppii myös tekemään itsenäisesti lyhyen koodin esim. lauseen ja numeroiden tulostaminen näytölle. Ilmiönä on siis tekstikoodin rakenne.

Opettaja selittää johdantona visuaalisen ja tekstipohjaisen koodin rakenteen vertailun ja antaa oppilaille koosteen tästä. Muuten tehtävät etenevät Tie koodariksi-sivuston seikkaperäisten esimerkkien kautta. Ohjelmointi on osa matematiikan OPS:ia, joten tehtäväkokonaisuus istuu hyvin matematiikan opetukseen. Tehtäväkokonaisuuden voi esimerkiksi mieltää vastaavan yhtä harjoitustyötä tai testiä. Oppilaat savat jälkikäteen itsearviointilomakkeen, jossa he voivat arvioida omaa oppimistaan.Tehtävät suoritetaan koulun kannettavilla tietokoneilla. Ohjeistus sisältyy tehtäväkokonaisuuteen.Aiempi osaaminen tulee esille siinä, että kokeneempien oppilaiden on helpompi edetä nopeampaan tahtia ja siten, kiitettävän arvosanan saavuttaminen on helpompaa. Tehtävät on kuitenkin tarkoitus tehdä opettajan määrittelemissä työpareissa. Tällöin kenenkään ei yksin tarvitse pohtia tehtäviä.

Algoritmista ajattelua voidaan edelleen hyödyntää matematiikan opiskelussa mm. hyödyntämällä GeoGebraa esimerkiksi funtiolaskennassa tai tasogeometriasssa.

Tuntisuunnitelma:
1. oppitunti (45 min): 30 min: Opettaja esittää koosteen visuaalisen ja tekstipohjaisen koodin rakenteesta. Käydään läpi muuttujat, käskyrakenteet, ehtolauseet, loopit, kirjastojen kutsuminen ja näytölle tulostaminen. Keskustellaan yhdessä näistä ja oppilaat voivat keksiä erilaisia tilanteita, missä koodia käytetään. 15 min tunnin lopussa: Oppilaat kirjautuvat Tie koodariksi-vivustolle. lukevat johdantokappaleen koodauksen historiasta ja tutustuvaat ensimmäisiin esimerkkeihin.

2.-4. oppitunti. Oppilaat etenevät itsenäisesti omissa ryhmissään, opettaja kannustaa eteenpäin ja tarvittaessa auttaa koodin rakentamisessa (opettajalla on oikeat koodit joka tehtävään).
4. oppitunti , 20 min tunnin lopussa: oppilaat täyttävät itsearviointilomakkeen.

Kappaleesen vaikuttavia voimia fysiikan simulaationa, 8.-9.lk

Reima Halmetoja, CC BY-SA 4.0

11.6.2020

Kappaleesen vaikuttavia voimia fysiikan simulaationa

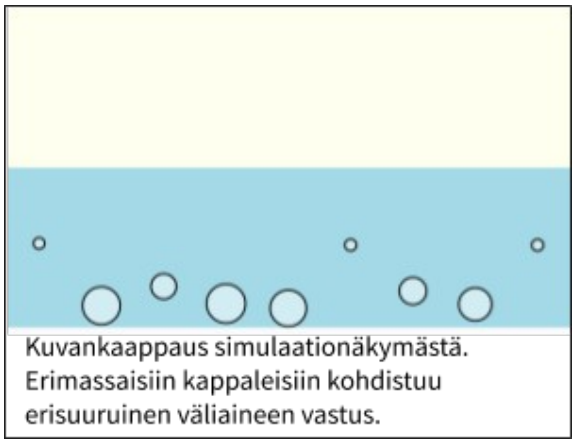
Reima Halmetoja

Johdanto ja aihepiiri

Tuntisuunnitelma käsittelee fysiikan oppimista ohjelmoinnin ja ohjelmasta tehtyjen havaintojen avulla. Ilmiötasolla tehtävässä esiintyy simulaatio nesteeseen putoavaan kappaleeseen kohdistuvista voimista. Jos tehtävän ohjelmointiosiota halutaan hahmottaa ilmiönäkökulmasta, käsittelee tehtävä ohjelmoinnillisena ilmiönä simulaatioita sekä simulaatioihin liittyvien tekijöiden (funktioiden ja niiden parametrien) muokkaamista.

Tehtävän tarkoitus käy parhaiten ilmi sen tavoitteista, jotka on listattu omaan osioonsa tässä suunnitelmassa.

Tuntisuunnitelma on suunniteltu 2x45 min tunnille. Oppilaille annettava ohjeistus on suunniteltu itseään eriyttäväksi. Kaikki oppilaat eivät siis tule ehtimään jokaista osiota. Toisaalta viimeistä osatehtävää voi tehdä mielivaltaisen kauan. Tehtävä on suunniteltu sellaiseksi, että keskeisimpiä tavoitteet tulee saavutettua tehtävänannon ensimmäisissä kohdissa.



Kuvankaappaus simulaationäkymästä. Erimassaisiin kappaleisiin kohdistuu erisuuruinen väliaineen vastus.

Ennakkovaatimukset ja kohderyhmä

- Opettajalla on oltava riittävä osaamispohja fysiikassa, keskeisenä aihealueena mekaniikka.
- Opettajalla olisi hyvä olla perustason osaaminen p5.js -kirjastolla täydennetystä JavaScript-ohjelmointikielestä, esimerkiksi laajuudessa, jolla yliopistojen ohjelmoinnin kaksi ensimmäistä kurssia käsittelevät ohjelmointia.
- Oppilaat tarvitsevat käyttöönsä tietokoneet, joissa on verkkoselain, joka suorittaa JavaScriptia.

- Tehtävä on suunnattu toteutettavaksi 8. tai 9. luokalla siinä vaiheessa kun fysiikassa ollaan käsittelemässä kappaleesen vaikuttavia voimia.
- Oppilailla on oltava fysiikan ennakkotietona käsitys seuraavista käsitteistä
- voima
- kappaleeseen kohdistuva painovoima
- väliaineen vastus
- noste
- Oppilailla on hyvä olla alkeistason ohjelmointiosaaminen komentorivipohjaisista ohjelmointikielistä, mielellään JavaScript tai jokin sen syntaktinen sukulainen (Processing, Java, c, jne.). Erityisesti oppilaiden tulisi tuntea ohjelmakoodin perusrakenne, funktiot ohjelmalohkoina, funktioiden kutsuminen sekä parametrien merkitys funktiota kutsuttaessa. Itse ohjelmointi on tehtävän alkuvaiheissa yksinkertaista koodin muokkaamista.
- Tehtävää varten on paras, jos käytössä olisi sähköinen oppimisympäristö, jonne oppilaille annettavan ohjeistuksen sekä oppilaiden palauttamat havainnot voitaisiin liittää. Näin arviointi on helpompaa ja oppimistehtävä voidaan liittää esimerkiksi osaksi oppilaan osaamisen portfoliota. Tehtävä voidaan kuitenkin muokata toteutettavaksi ilman oppimisympäristöä. Tässä dokumentissa oletetaan, että oppimisympäristönä käytössä on Google Classroom.

Tavoitteet

Fysiikka

- oppilas osaa tehdä havaintoja simulaatiosta
- oppilas tunnistaa putoamisliikkeessä oleviin kappaleisiin vaikuttavia voimia
- oppilas hahmottaa kappaleiden massan vaikutuksen niiden liikkeeseen eri tilanteissa
- oppilas hahmottaa kappaleeseen kohdistuvat voimat nesteessä ja voimien vaikutuksen kappaleen liikkeeseen
- oppilas osaa arvioida simulaatioiden rajoituksia.

Ohjelmointi ja algoritmit

- oppilas osaa tutkia valmista ohjelmakoodia ja ymmärtää sen toimintaa. Koodi on ohjelmakoodin rakenteen kannalta kehitysaskeleessa 4: aliohjelmien ja moduulien rakentaminen sekä parametrien välittäminen
- oppilas muodostaa käsityksen ohjelmakoodin avulla luodun simulaation mahdollisuuksista fysiikan havainnointi- ja mallinnusvälineenä

- oppilas osaa muokata valmista ohjelmakoodia sekä osaa kokeilla muokkauksien vaikutusta ohjelmakoodin toimintaan
- lisäksi: useita ohjelmointiin liittyviä käsitteitä ja rakenteita, riippuen oppijoiden lähtötasosta. Ohjelmakoodi on kuitenkin sopivalla ohjauksella riittävällä tasolla ymmärrettävissä, jotta tehtävänannon mukaiset havainnot voidaan tehdä ja fysiikan oppimistavoitteet voidaan saavuttaa.

Osaamisen arviointi

Oppimistehtävä on jaettu vaiheittain eri askeleisiin, joissa edetään pienemmästä vaatimustasosta suurempaan. Etenemisessä on pyritty pitämään mielessä fysiikan käsiterakenteen muodostuminen ja osaamistaksonomiatyylinen ajattelu, jossa edetään havainnoista ja toistosta soveltamiseen, arviointiin ja uuden luomiseen.

Tehtävän formatiivisessa arvioinnissa keskitytään oppimisen ohjaamiseen ja palautteen antamiseen. Formatiivinen arviointi toteutetaan siten, että opettaja kiertää ohjaamassa oppilaita tehtävässä ja antamassa palautetta.

Jos tehtävä halutaan arvioida summatiivisesti, voidaan tarkastella oppilaan etenemistä ohjeistuksessa. Etenemistä ohjeistuksessa peilataan tehtävän oppimistavoitteisiin. Arvioinnissa arvioidaan siis sitä, miten yllä olevat tavoitteet toteutuvat oppilaan osalta. Tavoitteet voivat toteutua osaamiseltaan hyvin eri tasoisesti, esim. tavoite oppilas osaa muokata valmista ohjelmakoodia sekä osaa kokeilla muokkauksien vaikutusta ohjelmakoodin toimintaan toteutuu eri tasolla, jos oppilas löytää sopivan muokattavan parametrin tai jos oppilas osaa ohjelmoida uuden voiman simulaatioon.

Oppilas saa tietyssä mielessään palautetta osaamisestaan, koska hän välttämättä havaitsee etenemisensä tehtävän ohjeistuksen mukaisesti.

Oppilaiden ohjeistaminen -osiossa käsitellään tunnin jälkeen pidettävää purkukeskustelua, jossa oppilaita ohjeistetaan itsearviointiin pohtimalla mitä ovat oppineet tehtävässä yhteisessä purkukeskustelussa.

Työskentelyvälineet ja opetusmenetelmät

Tehtävässä oppilaat etenevät pienryhmissä ja tekevät fysiikan havaintoja sekä ohjelmoivat ohjeistuksen (ns. työkortin) mukaan.

Oppilaat käyttävät tehtävässä ohjelmointiympäristöä, joka toimii selaimessa osoitteessa editor.p5js.org.

Tehtävässä oppilaat tekevät havaintoja ohjelmoidusta fysiikan simulaatiosta. Simulaation avulla pyritään antamaan mahdollisus oppilaalle vaikuttaa suoraan fysiikan lainalaisuuksiin. Oppilaita

pyritään motivoimaan tehtävään siten, että he pystyvät vaikuttamaan simulaatioon ja näkevät heti oman työnsä tulokset. Oppilaat työskentelevät myös vertaisryhmissä, jolloin tehtävässä on mukana jonkinasteinen yhteisöllisyys. Alussa oppilaita ohjeistetaan tarkasti, mutta pidemmälle tehtävässä edenneille annetaan autonomiaa.

Oppilaille annettava ohjeistus on hyvä siirtää oppimisympäristöön kuten Google Classroom. Oppimisympäristössä jokaiselle oppilaalle jaetaan dokumentti, jossa on varsinainen ohjeistus (työkortti), johon hän tekee muistiinpanot havainnoista ja liittää mahdollisia kuvankaappauksia.

Ohjeistuksen lisäksi oppimisympäristöön on liitettävä simulaattorin ohjelmakoodi, joka on tämän dokumentin liitteenä. Ohjelmakoodi on muokattu p5js.org:in esimerkkikoodista [1], joka puolestaan pohjautuu Daniel Schiffmanin The Nature of Code-kirjaan [2], jossa on kosolti fysiikan simulaatioihin käyttökelposta materiaalia Processing-ohjelmintikielellä

Tehtävän ohjeistus

Ennen osaamistehtävää oppilaiden kanssa käsitellään fysiikassa kappaleesen vaikuttavia voimia: painovoima, noste ja väliaineen vastus. Fysiikan ennakkotietoja käsittelevän osuuden ja simulaatio-osuuden suhde on hyvä suunnitella etukäteen oppilasryhmän mukaisesti.

Simulaatiotehtävän alussa oppilaat jaetaan 2-3 hengen ryhmiin. Oppilaita rohkaistaan auttamaan toisia, jos jollakulla ryhmässä on vaikeuksia ohjelmakoodinsa kanssa.

Oppilaita ohjeistetaan avaamaan tehtävänanto oppimisympäristössä.

Jokainen oppilas ohjeistetaan täyttämään oman raportin, joka tehdään suoraan vastauksina oppilaille annettuun ohjeistukseen (ks. alla). Oppilasryhmät etenevät itsenäisesti ohjeistuksen (työkortin) mukaan. Tehtävä on eriytetty etenemään ohjeistuksen avulla siten, että kaiken tasoisten oppilaiden pitäisi saada siitä onnistumisen ja oppimisen kokemuksia, mutta myös taitavimmille pitäisi löytyä tekemistä. Halutessaan opettaja voi poistaa tehtäviä helpottavat virheet.

Viimeisissä tehtävissä pitää ohjelmoida noste. Eräs mahdollinen ratkaisu nosteen ohjelmoimiseksi on liitteessä 2.

Tehtävän purku

Tunnin lopuksi lopetetaan ohjelmointi ja puretaan tehtävä. Tehtävän purun tavoite on toimia välineenä oman oppimisen reflektointiin ja itsearviointiin. Oppilaat sekoitetaan siten, että jokaisesta ryhmästä 1-2 oppilasta menee toiseen ryhmään, siten että muodostuu uudet 2-3 henkilön ryhmät, joissa kaikki ovat toisilleen tehtävän kannalta uusia. Oppilaat ohjeistetaan kertomaan ryhmissä muille (1) mihin kohtaan he ehtivät tehtävässä, (2) mikä tehtävässä oli helppoa ja mikä oli vaikeaa ja (3) mitä he oppivat tehtävästä.

Porinatuokion jälkeen keskustelu voidaan siirtää koko luokkahuoneen keskusteluksi, jossa kysytään mitä oppilaat ajattelevat simulaatioista fysiikan havainnointimenetelmänä. Opettaja voi ohjata keskustelua siihen, missä muissa tilanteissa valmiin ohjelmakoodin muokkaaminen ja siihen lisäyksien tekeminen on hyödyllistä.

Ohjeistus oppilaille (työkortti)

1. Avaa selain ja mene osoitteeseen editor.p5js.org, jolloin saat käyttöösi editorin p5.js -ohjelmaasi varten.
2. Kopioi oppimisympäristössä oleva simulaattorin ohjelmakoodi p5.js -editori-ikkunaasi.
3. Aja ohjelmakoodi. Voit suorittaa ohjelmakoodin moneen kertaan nollaamalla tilanteen klikkaamalla ruutua. Kirjaa ylös minkälaisia voimia havaitset vaikuttavan kappaleisiin.
4. Etsi ohjelmakoodista kohta, jossa määritellään Kappale-oliot. (function Kappale(m, x, y)). Minkälaisia ominaisuuksia Kappale-olioilla on tässä simulaatiossa?
5. Piirrettyjen kappaleiden koot ovat suoraan verrannollisia niiden massoihin (voit etsiä koodista kohdan, jossa kappaleet piirretään circle()-funktiolla varmistuaksesi tästä).

Miten kappaleen massa vaikuttaa sen putoamiskiihtyyvyteen?

6. Miten simulaation perusteella kappaleen massa vaikuttaa siihen kohdistuvaan väliaineen vastukseen?
7. Muokkaa ohjelmakoodia siten, että kappaleiden massat ovat suurempia. Vihje: etsi koodi kappaleet[i] = new Kappale(0.4*(i+1), 40 + i * 70, 0); ja kokeile arvojen muokkaamista.

Miten muokkauksesi vaikuttavat tekemiisi havaintoihin?

8. Vaihda kappaleiden massat taas alkuperäisiksi (kappaleet[i] = new Kappale(0.4*(i+1), 40 + i * 70, 0);)
9. Muokkaa ohjelmakoodia siten, että väliaineen vastuksen kappaleisiin kohdistama voima on suurempi. Kokeile useita eri arvoja.

Kaikilla fysiikan malleilla on rajoituksia. Tämän simulaation avulla voidaan havainnollistaa painovoiman ja väliaineen vastuksen vaikutusta erimassaisiin kappaleisiin. Simulaatiolla on kuitenkin omat rajoituksensa, jonka jälkeen se ei kuvaa tilannetta hyvin.

10. Minkälaisessa tilanteessa havaitset selvästi, että kyseinen simulaatio ei mallinna tapahtumaa uskottavasti?
11. Palauta kappaleeseen kohdistuva väliaineen vastus alkuperäisen suuruiseksi.

12. Simulaatio mallintaa tasan kahta voimaa, jotka kohdistuvat nesteessä olevaan kappaleeseen: painovoima ja väliaineen vastus.

Sen sijaan, että kasvatettaisiin väliaineen vastusta, mikä voima simulaatioon pitäisi ohjelmoida, jos haluttaisiin mallintaa nestettä vähemmän tiheämpien kappaleiden käyttäytymistä nesteessä?

13. Tutki koodia, jossa kappaleeseen kohdistetaan painovoima ja ehtoa, jossa kappaleeseen kohdistuu väliaineen vastus. Ohjelmoi näiden perusteella kappaleeseen vaikuttava puuttuva voima lisäämällä sopivaan suuntaan vaikuttava voimavektori sopivaan kohtaan ohjelmakoodia.

Vihje: voimavektorin parametreistä ensimmäinen tarkoittaa liikettä positiivisen x-koordinaatin ja toinen positiivisen y-koordinaatin suuntaan.

14. Muokkaa kappaleesen kohdistuvan nosteen voimavektorin suuruutta siten, että kappale a) kelluu, b) leijuu, c) uppoaa.

15. Keksi kokeiltavia asioita ohjelmakoodiin, kokeile muokkauksien vaikutuksia simulaatioon ja kirjaa ylös miten muokkaukset vaikuttavat tilanteeseen fysiikan kannalta ja mitä havaintoja teet simulaatiosta.

Liitteet

Liite 1: tehtävänannon ohjelmakoodi

Koodi on suomennettu p5js.org:in englanninkielisestä voimia käsittelevästä esimerkkikoodista [1], jotta se olisi suomenkieliselle ymmärrettävämpi. Lisäksi koodista on poistettu satunnaisuus alustettavien kappaleiden massoihin liittyen.

Alkuperäisen ja tämän koodin lisenssi: https://creativecommons.org/licenses/by-nc-sa/4.0/

```
 // Ohjelmakoodi havainnollistaa eri voimien vaikutusta
// kappaleisiin (Kappale-olioita)
// Painovoima vaikuttaa kappaleisiin jatkuvasti
// kappaleisiin kohdistuu väliaineen vastus, kun ne ovat "vedessä"
// Viisi liikkuvaa Kappale-oliota, jotka alustetaan taulukkoon
let kappaleet = [];
```

```
// Neste-olio, jolla mallinnetaan väliaineen vastus
let neste;
function setup() {
  createCanvas(640, 360);
  reset();
  // Luodaan Neste-olio
  neste = new Neste(0, height / 2, width, height / 2, 0.1);
}
function draw() {
  background("ivory");
  // Piirretään vesi
  neste.display();
  for (let i = 0; i < kappaleet.length; i++) {
    // Liikkuuko Kappale Nesteessä?
    if (neste.contains(kappaleet[i])) {
      // Lasketaan väliaineen vastus
      let vastusVoima = neste.laskeVastusVoima(kappaleet[i]);
      // Kohdistetaan vastusvoima liikkuviin kappaleisiin
      kappaleet[i].applyForce(vastusVoima);
    }
    // Massan suuruus vaikuttaa painovoimaan tässä
    let painovoima = createVector(0, 0.1 * kappaleet[i].massa);
    // Kohdistetaan painovoima kappaleisin
    kappaleet[i].applyForce(painovoima);
    // Päivitetään ja piirretään kappaleet
    kappaleet[i].update();
    kappaleet[i].display();
    kappaleet[i].checkEdges();
  }
}
// Nollataan tilanne, jos hiirtä klikataan
function mousePressed() {
  reset();
}
```

```
// Alustetaan erimassaiset Kappale-oliot
function reset() {
  for (let i = 0; i < 9; i++) {
    kappaleet[i] = new Kappale(0.4*(i+1), 40 + i * 70, 0);
  }
}
let Neste = function(x, y, w, h, c) {
  this.x = x;
  this.y = y;
  this.w = w;
  this.h = h;
  this.c = c;
};
// Onko Kappale Nesteessä?
Neste.prototype.contains = function(m) {
  let l = m.sijainti;
  return l.x > this.x && l.x < this.x + this.w &&
         l.y > this.y && l.y < this.y + this.h;
};
// Lasketaan väliaineen vastus
Neste.prototype.laskeVastusVoima = function(m) {
  // Suuruus on vakiokerroin * nopeuden neliö
  let nopeus = m.vauhti.mag();
  let vastuksenSuuruus = this.c * nopeus * nopeus;
  // Suunta on vastakkainen kuin vauhti
  let vastusVoima = m.vauhti.copy();
  vastusVoima.mult(-1);
  // Skaalaa suuruuden mukaan
  vastusVoima.normalize();
  vastusVoima.mult(vastuksenSuuruus);
  return vastusVoima;
};
Neste.prototype.display = function() {
  noStroke();
```

```
  fill("lightblue");
  rect(this.x, this.y, this.w, this.h);
};
function Kappale(m, x, y) {
  this.massa = m;
  this.sijainti = createVector(x, y);
  this.vauhti = createVector(0, 0);
  this.kiihtyvyys = createVector(0, 0);
}
// Newtonin II laki: F = m * a
// tai a = F / m
Kappale.prototype.applyForce = function(force) {
  let f = p5.Vector.div(force, this.massa);
  this.kiihtyvyys.add(f);
};
Kappale.prototype.update = function() {
  // vauhti muuttuu kiihtyvyyden perusteella
  this.vauhti.add(this.kiihtyvyys);
  // vauhti vaikutta sijaintiin
  this.sijainti.add(this.vauhti);
  // kiihtyvyys pitää nollata jokaisessa ruudussa
  this.kiihtyvyys.mult(0);
};
Kappale.prototype.display = function() {
  stroke(0);
  strokeWeight(2);
  fill(255,127);
  circle(this.sijainti.x, this.sijainti.y, this.massa * 16);
};
// Pompauta kappaleet ruudun alalaidasta
Kappale.prototype.checkEdges = function() {
  if (this.sijainti.y > (height - this.massa * 8)) {
    // Lisätään hieman vaimenemista pomppauksessa
    this.vauhti.y *= -0.9;
```

```
    this.sijainti.y = (height - this.massa * 8);
  }
};
```

Liite 2: draw() -funktio, joka implementoi nosteen, joka on suurempi kuin kappaleeseen kohdistuva painovoima.

Koodi on muokattu p5js.org:in esimerkkikoodista [1].

Alkuperäisen ja tämän koodin lisenssi: https://creativecommons.org/licenses/by-nc-sa/4.0/

```
function draw() {
  background("ivory");
  // Piirretään vesi
  neste.display();
  for (let i = 0; i < kappaleet.length; i++) {
    // Alla olevalla rivillä kerroin -0.2 kuvaa kappaleeseen kohdistuvan nosteen
ja kappaleen painon (painovoiman) suhdetta.
    let noste =  createVector(0, -0.2 * kappaleet[i].massa);

    // Liikkuuko Kappale-olio nestessä?
    if (neste.contains(kappaleet[i])) {
      // Lasketaan väliaineen vastus
      let vastusVoima = neste.laskeVastusVoima(kappaleet[i]);
      // Kohdistetaan vastusvoima liikkuviin kappaleisiin

      kappaleet[i].applyForce(vastusVoima);
      kappaleet[i].applyForce(noste);

    }
    // Massan suuruus vaikuttaa painovoimaan tässä
    let painovoima = createVector(0, 0.1 * kappaleet[i].massa);
    // Kohdistetaan painovoima kappaleisin
    kappaleet[i].applyForce(painovoima);
    // Päivitetään ja piirretään kappaleet
```

```
    kappaleet[i].update();
    kappaleet[i].display();
    kappaleet[i].checkEdges();
  }
}
```

Viitteet

[1] p5.js-ohjelmakirjaston esimerkit: https://p5js.org/examples/simulate-forces.html <haettu 1.3.2020>

[2] Shiffman, D. (2020). The Nature of Code. Saatavissa: https://natureofcode.com/ <haettu 1.3.2020>

Pythagoraan lause, 8.-9. lk

9.11.2022

1 Pythagoraan lauseen ja ehtorakenteen harjoittelua Pythonilla

Tällä oppitunnilla oppilaat harjoittelevat Pythagoraan lausetta ohjelmointia hyödyntäen. Samalla opitaan käyttämään ehtolausetta lausekielisessä ohjelmoinnissa. Tunti soveltuu pidettäväksi 8. tai 9. luokalla, kun Pythagoraan lausetta käsitellään. Aikaa on suunniteltu käytettävän kaksoistunnin (2*45 min) verran. Valitsin aiheen, koska Pythagoraan lause on yksi tärkeistä asioista yläkoulun matematiikassa ja sitä voidaan hyödyntää monissa soveltavissa tehtävissä sekä todellisessa elämässä. Se soveltuu hyvin yhdistettäväksi ohjelmointiin ja tämän esimerkkitunnin lisäksi aihe tarjoaisi paljon muitakin ohjelmointimahdollisuuksia. Ohjelmoinnin sisällöistä valitsin ehtorakenteen, jotta oppitunnin voi pitää pienin pohjatiedoin, mutta ehtorakenteen avulla saa silti aikaan mielekkäitä koodeeja, joita voidaan testata ja hyödyntää mm. kirjan tehtäviä laskemalla.

1 Esitiedot

Oppilaiden olisi hyvä osata Pythagoraan lauseen määritelmä sekä kateetin ja hypotenuusan pituuden laskeminen kahden muun sivun pituuden perusteella. Tunti on suunniteltu Pythagoraan lauseen osaamisen vahvistamiseen pikemmin kuin opettamiseen. Ohjelmointia Pythonilla olisi hyvä olla kokeiltu ainakin yhden oppitunnin verran ja muuttujien käyttö sekä print-käsky tulisivat olla tuttuja.

2 Oppimistavoitteet

Oppitunnin ohjelmointitavoitteena on ehtorakenteen opettelu. Matematiikan tavoitteena on vahvistaa Pythagoraan lauseen osaamista. Tavoitteena on tuottaa ohjelmakoodit, joilla pystyy todentamaan onko jokin kolmio suorakulmainen tai laskemaan kahden annetun sivun pituuden perusteella suorakulmaisen kolmion kolmannen sivun, joka voi olla kateetti tai hypotenuusa. Kolmion sivun ratkaiseminen Pythagoraan lauseella tulee suoraan opetussuunnitelman perusteista. Ohjelmoinnista mainitaan opetusssunnitelman perusteissa ehto- ja toistorakenteiden käyttö, joista tässä harjoitellaan ehtolausetta.

3 Työvälineet

Ohjelmointiin käytetään Python-ohjelmointikieltä replit.com-ympäristössä. Oppilailla tulisi olla joko omat tietokoneet/Chromebookit (tai laite aina kahta oppilasta kohden). Työskentelyyn soveltuu myös jokin muu Pythopn-editori. Replit:n hyvä puoli on, se on selainpohjainen ja oppilaat voivat kirjautua sinne olemassaolevilla koulun tunnuksillaan. Kirjautuminen mahdollistaa koodien tallentamisen ja jatkamisen seuraavilla opetuskerroilla. Lisäksi oppilailla olisi hyvä olla käytössään tunnilla paperia ja kynä. Opettajalla tulee olla käytössään koneen ja Replit-ympäristön lisäksi väline, jolla heijastaa koodia taululle, tai hänen tulee soveltaa ohjeita tauululle kirjoitettaviksi.

4 Arviointi

Arvionnin kohteena matematiikassa ovat

- Pythagoraan lauseen käyttö kolmion suorakulmaisuuden testaamiseen
- hypotenuusan pituuden laskeminen Pythagoraan lauseen avulla
- kateetin pituuden laskeminen Pythagoraan lauseen avulla

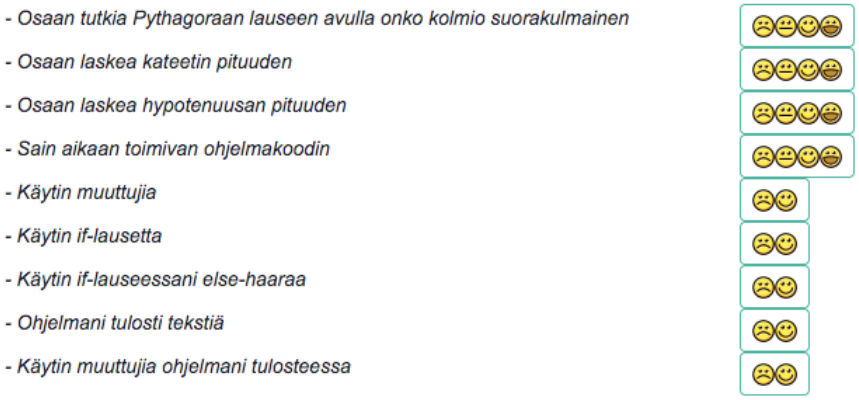
Arvioinnin kohteena ohjelmoinnissa ovat

- toimivan ohjelmakoodin aikaansaaminen
- muuttujien käyttö
- ehtolauseen käyttö (if-lause)
- ehtolauseessa sen huomioiminen, ettei ehto täyty (else)
- mielekäs tuloste, jossa kirjoitusta
- muuttujien arvot mukana tulosteessa

Valitsisin itse arviointitavaksi itsearvioinnin, jonka oppilaat tekisivät tunnin lopussa joko sähköisesti tai paperilla. Opettaja lisäksi havainnoi oppilaiden etenemistä ja osaamista oppituntien aikana sekä kysyy työskentelyä eteenpäin ohjaavia kysymyksiä.

Itsearvioinnin voi toteuttaa esimerkiksi Qridissä, jolloin se voi näyttää oppilaalle esimerkiksi tältä:

Itsearvioitavat väitttämät voivat olla esimerkiksi seuraavat



Halutessaan opettaja voi pyytää oppilaita arvioimaan myös muita asioita kuten vuorovaikutusta ja yhteistyötä toisten oppilaiden kanssa.

5 Ohjelmointi

Oppitunnin alussa opettajan tulee varmistaa, että kaikilla oppilailla tai pareilla on toimiva kone ja että he pääsevät kirjautumaat replit-ympäristöön sekä saavat luotua uuden projektin. Tarvittaessa opettaja voi näyttää tämän yhteisesti. Ajatuksena oppitunnilla on, että jokainen oppilas kirjoittaa omat ohjelmakoodinsa, mutta oppilaita rohkaistaan temään yhteistyötä sekä auttamaan toinen toisiaan kaikissa vaiheissa.

1 Osa 1: Ehtolauseen esittely

Opettaja esittelee if-lauseen idean yksinkertaisen esimerkin avulla. Kaikki esimerkit on hyvä olla opettajalla valmiina Replit-ympäristössä, josta hän voi esitellä ne esimerkiksi älytaulua tai projektoria käyttäen.

Esimerkki yksinkertaisesta if-lauseesta:

```
x=-4 # Annetaan arvo muuttujalle x
if x<0: # Jos x on nollaa pienempi tulostetaan allaoleva lause
  print('Luku on negatiivinen')
else: # Muussa tapauksessa tulostetaan seuraava lause
  print('Luku on positiivinen tai nolla, eli luonnollinen luku')
```

Halutessaan opettaja voi samalla näyttää peräkkäisten if-lauseiden käytön (elif), mutta se ei ole tunnin ohjelmointitehtävissä tarpeen

Oppilaiden tehtävänä on kopioida koodi itselleen ja kokeilla miten muuttujien arvojen, ehdon sekä tulosteiden muutokset vaikuttavat. Oppilaat keskustelevat havainnoistaan toistensa kanssa ja keskeisiä havaintoja voidaan käydä yhteisesti.

2 Osa 2: Ensimmäinen ohjelmointitehtävä, kolmion suorakulmaisuuden testaaminen

Kun kaikki ovat saaneet if-lauseen toimimaan, voidaan siirtyä ensimmäiseen ohjelmointitehtävään. Tehtävänä on kirjoittaa koodi, joka testaa Pythagoraan lauseen avulla onko kolmio suorakulmainen, kun tiedetään kaikkien sivujen pituudet.

Tässä vaiheessa oppilaille esitellään merkinnät yhtäsuuruudelle (==), kertomiselle (*) sekä potenssille (**) tai kerrataan ne (oppilas voi valintansa mukaan käyttää potenssia tai kertolaskua, sillä a**2 voidaan kirjoittaa myös a*a). Oppilaiden tulisi itse pystyä palauttamaan mieleen Pythagoraan lause.

Oppilaiden tehtävänä on

- miettiä mitä muuttujia he tarvitsevat
- kirjoittaa if-lauseeseen ehdoksi Pythagoraan lause muuttujia käyttäen
- kirjoittaa else-osa
- kirjoittaa mielekäs tuloste/tulosteet.
  Apuna he voivat käyttää ensimmäistä esimerkkiä.

Esimerkki ohjelmakoodista:

```
1 # Tutkitaan Pythagoraan lauseen avulla onko kolmio suorakulmainen, kun tiedetään
  kolmion sivujen pituudet
2 a=3 # Annetaan arvo kateetille a
3 b=4 # Annetaan arvo kateetille b
4 c=6 # Annetaan arvo hypotenuusalle c
5 if a**2+b**2==c**2: # Tutkitaan päteekö Pythagoraan lause. Jos pätee tulostetaan
  allaoleva
6   print('Kolmio on suorakulmainen')
7 else: # Muussa tapauksessa tulostetaan seuraava lause
8   print('Kolmio ei ole suorakulmainen')
```

Tarvittaessa opettaja voi antaa oppilaille avuksi pelkät kommenttirivit, joiden perusteella oppilaat voivat kirjoittaa varsinaisen koodin.

3 Osa 3: Toinen ohjelmointitehtävä, hypotenuusan laskeminen

Kun oppilas on saanut edellisen koodin toimimaan, voi hän siirtyä kirjoittamaan koodia hypotenuusan laskemiseksi. Ensin oppilaan olisi hyvä muodostaa lauseke hypotenuusan laskemiseksi paperilla. Oppilas voi käyttää laskussaan joko muuttujia tai konkreettisia arvoja. On myös mahdollista hyödyntää valmista lauseketta, jos oppilaalla on sellainen esimerkiksi muistiinpanoissaan tai kirjassaan. Tarvittaessa opettaja voi auttaa oppilaita tässä vaiheessa, mikäli asia ei ole heille vielä selvä.

Neliöjuuren laskemiseksi Python-ohjelmassa tarvitaan avuksi Pythonin kirjasto math. Kun ensimmäiset oppilaat siirtyvät tähän vaiheeseen, voi opettaja laittaa taululle esimerkin, jossa on tarvittavat esimerkit
import.math
math.sqrt(81) #laskee luvun 81 neliöjuuren, englanniksi square root

Kun oppilailla on tiedossa lauseke hypotenuusan laskemiseksi sekä keino neliöjuuren laskemiseksi, voivat he kirjoittaa koodin samaan tapaan kuin edellisessä tehtävässä pohtien tarvittavat muuttujat, ehtolauseet else-haaroineen sekä tulosteet.

Esimerkki ohjelmakoodista:

```
# Lasketaan hypotenuusan pituus Pythagoraan lauseella
import math # Tuodaan math-kirjasto käyttöön
a=10 # Annetaan arvo kateetille a
b=12 # Annetaan arvo kateetille b
c=math.sqrt(a**2+b**2) # Lasketaan hypotenuusan c pituus Pythagoraan lauseella
print('Kun suorakulmaisen kolmion toinen kateetit ovat',a,'ja',b,'niin
hypotenuusan pituus on',c)
```

4 Osa 4: Kolmas ohjelmointitehtävä, kateetin laskeminen

Kolmantena ja viimeisenä ohjelmointitehtävänä oppilaiden tulee kirjoittaa koodi kateetin laskemiseksi. Koodi kirjoitetaan vastaavasti kuin hypotenuusaa laskiessa, selvittäen ensin paperilla kaavan kateetin laskemiseksi ja siirtyen vasta sen jälkeen koodin kirjoittamiseen. Tehtävässä voi hyödyntää edellisen tehtävän koodia, josta saa pienillä muutoksilla muokattua tämän tehtävän ratkaisun.

Esimerkki ohjelmakoodista:

```
# Lasketaan kateetin pituus Pythagoraan lauseella
import math # Tuodaan math-kirjasto käyttöön
a=4 # Annetaan arvo kateetille a
c=5 # Annetaan arvo hypotenuusalle c (oltava suurempi kuin kateetin arvo)
b=math.sqrt(c**2-a**2) # Lasketaan kateetin b pituus Pythagoraan lauseella
print('Kun suorakulmaisen kolmion toinen kateetti on',a,'ja hypotenuusa',c,'niin
toisen kateetin pituus on',b)
```

5 Osa 5: Lisätehtävä edistyneille

Oppilaat, joilla on ohjelmointikokemusta tai jotka suoriutuvat tehtävistä nopeasti, voivat lisätä ohjelmiinsa vuorovaikutteisuutta käyttämällä input-komentoa. Heille voi esitellä komennon, sekä miten merkkijonona saatu syöte muutetaan luvuksi int-komennolla. Kaksi jälkimmäistä ohjelmaa kannattaa yhdistää yhdeksi ohjelmaksi, jossa on omat ehtolauseen

haarat hypotenuusan sekä kateetin laskemiseksi. Kumpaa kulloinkin lasketaan, voi kysyä käyttäjältä.

Esimerkki vuorovaikutteisesta koodista ensimmäiseen ohjelmointitehtävään, jossa tutkittiin onko kolmio suorakulmainen:

main.py

```
#Kysytään käyttäjältä kolmion sivut, joiden perusteella lasketaan, onko kolmio suorakulmainen
print('Tervetuloa selvittämään, onko kolmio suorakulmainen!')
c = input('Kuinka pitkä on kolmion pisin sivu?')
a = input('Kuinka pitkä on kolmion toiseksi pisin sivu?')
b = input('Kuinka pitkä on kolmion lyhin sivu?')
c = int(c)  #muutetaan merkkijonona saadut syötteet luvuiksi
a = int(a)
b = int(b)
if a**2 + b**2 == c**2:  #testatann toteutuuko Pythagoraan lause
  print('Kolmio, jonka sivut ovat',b,',',a,'ja',c,'on suorakulmainen')
else:
  print('Kolmio, jonka sivut ovat',b,',',a,'ja',c,'ei ole suorakulmainen')

```

Python-ohjelmointia 9.lk

Nimetön,

11.6.2020

Python-ohjelmointia 9. luokkalaisille

(23 oppilasta, ohjelmoinnin osaaminen vähäistä)

Aiheen valinta:

Ops:n ohjelmoinnin tavoitteiden saavuttaminen. Ohjelmointia itsenäisenä aiheena, ei varsinaisesti sovelleta toiseen oppiaineeseen.

Arvioinnin kohteet:

- Tehtävien eteneminen

- Aktiivisuus

- Tunnin lopuksi itsearvio: mitä opin? (tieto/taito)

Tavoitteet:

- Ohjelmointikieli tutuksi

- Onnistumisen iloa

- Joku käsitys ohjelmien takana olevasta maailmasta.

- Joku innostuisi aiheesta enemmänkin.

1. *tunti:* Alustusta ohjelmointiin, kirjautuminen tie koodariksi -sivustolle ja pari tehtävää

- Oppilaat kutsutaan tunnin aluksi Teams-kokoukseen. Ovat käyttäneet sitä koko etäopetuksen ajan.

- Tutustutaan ensin tietokoneen kuvakaappaus-työkaluun. Kiinnitetään se tehtäväpalkkiin. Sillä on helppo ottaa näytöstä kuvia opettajalle tai kaverille, jos haluaa kysyä tehtävästä tai apua virheen etsintään.

- Keskustelua ohjelmoinnista. Mitä on ohjelmointi ja missä siihen törmää päivittäin? Mitä muistavat viime kevään turtle-ohjelmoinnista. Katsotaan esimerkki html-koodista oman koulun kotisivulta (https://www.tampere.fi/varhaiskasvatus-ja-koulutus/esiopetus-ja-perusopetus/koulut/kammenniemen-koulu.html)

- Mennään sivulle (linkki annetaan tiimin viestit osiossa):

    https://tie.koodariksi.fi/alkeet/

i. Klikkaa: Kirjaudu sisään -painikkeesta oikeasta yläreunasta

ii. Luodaan tunnukset: Eikö sinulla ole vielä tunnusta? Voit luoda sen tästä.

iii. Tee tunnus, josta sinua ei voida tunnistaa. Lue tietosuojaselostus. Täytä myös muut kohdat, Laita nimesi kohdalle etunimi. Luo käyttäjä.

iv. Tämän jälkeen liity ryhmään avaimella [sijoita avain tähän]

v. Laita opettajalle viestiä, kun olet tässä vaiheessa. Voit odotellessa tutustua materiaaliin.

- Opettaja jakaa oppilaat ryhmiin, joissa heidän on tarkoitus etänä tehdä tehtäviä yhdessä neuvoen ja kysyen. Ryhmät 2-3 henkeä ja mahdollisimman tasaiset, mutta kuitenkin niin, että työskentely sujuu.

- Kun kaikki ovat valmiita, tutustutaan yhdessä kahteen ensimmäiseen lukuun. Käydään läpi (Ohjelmoinnin historiaa ja ensimmäinen ohjelma) ja seuraavat asiat: #, print, sulkujen käyttö, lainausmerkkien käyttö (merkkijono), laskutoimitusmerkit (erityisesti *), pilkkujen käyttö.

- Tämän jälkeen oppilaat tekevä luvusta 2 tehtävät 1-4.

-Tunnin lopuksi itsearviointi Teams:n tehtävissä olevaan päiväkirjaan.

2. *tunti*: Muuttujat ja toistaminen

- Kerrataan yhdessä ensimmäisen tunnin asioista: #, print, sulkujen käyttö, lainausmerkkien käyttö (merkkijono), laskutoimitusmerkit (erityisesti *), pilkkujen käyttö.

- Käydään yhdessä läpi muuttuja, arvon sijoittaminen muuttujaan, muuttujan arvon vaihtuminen, laskutoimitukset

- Oppilaat tekevät edellisen tunnin ryhmissä luvun 3 tehtävät. Jos joku oppilas on nopea, hän voi jo mennä tutustumaan lukuun 4 (ja jos on ymmärrystä enemmänkin, voi edetä omatoimisesti).

- Kuitenkin vasta, kun kaikki ovat saaneet 3. luvun tehtyä, käymme yhdessä läpi luvun 4 asiat: for-silmukka, sisennys koodissa, range(a), range(a,b), range(a,b,c)

- Oppilaat tekevät luvun 4 tehtävät. Osan kanssa pitänee tehdä yhdessä, joten pidämme videokokouksen auki niille ryhmille, jotka epäilevät etteivät saa tehtävää tehtyä keskenään. Jos joku on vallan nopea, hän voi mennä omatoimisesti eteenpäin.

- Tunnin lopuksi itsearviointi Teams:n tehtävissä olevaan päiväkirjaan.

- Tarkoitus jatkaa edelleen seuraavilla tunneilla ja ohjata kiinnostuneet harrastamaan/kursseille.

Klassinen ja tilastollinen todennäköisyys Pythonilla, 9. lk

Saana Saajoranta, CC BY-SA 4.0

17.10.2021

Klassinen ja tilastollinen todennäköisyys Pythonilla 9.lk

Aihepiirin valinta ja rajaus:
Valitsin tuntisuunnitelman aiheeksi todennäköisyyslaskennan Pythonin avulla, jotta oppilaat pääsevät soveltamaan ohjelmointia matematiikan opiskelun yhteydessä. Tuntisuunnitelma rajataan koskemaan klassista ja tilastollista todennäköisyyttä, mutta voi olla laajennettavissa koskemaan peräkkäisten tapahtumien todennäköisyyttä. Pythonin osalta tässä tuntisuunnitelmassa keskitytään laatimaan ohjelma, joka arpoo satunnaisen luvun valitulta väliltä.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi:
Tavoitteena on tutkia satunnaisia ilmiöitä ohjelmoinnin avulla ja simuloida tietokoneella toistokertoja, joiden avulla saataisiin ilmiön käyttäytymisestä analysoitavaa dataa. Matematiikan osalta tavoitteena on OPS:n mukaiset todennäköisyyslaskennan tavoitteet, jotka ohjaa oppilasta määrittämään tilastollisia tunnuslukuja ja laskemaan todennäköisyyksiä.

Arviointi tapahtuu tehtävien tekemisen ja itsearvioinnin avulla. Tunnin lopussa oppilas vastaa itsearviointiin ja keskustellaan oppilasryhmän kanssa ohjelmoinnin soveltamisesta todennäköisyyslaskennassa.

Työskentelyvälineet ja opetusmenetelmät:
Oppilaiden oletetaan opiskelleen ohjelmointia Pythonilla (selaimella tai koneelle ladatulla ohjelmalla) ja hallitsevat ainakin perusasiat, kuten yksinkertaisen koodin laatimisen ja sen osat (muuttujat, toistamisen,..). Lisäksi oppilaiden tulee hakea ohjelmaa varten kirjastosta satunnaisuuteen liittyvä random-apumoduuli. Tässä kohtaa oletetaan kirjastosta työkalujen hakemisen tutuksi import -toiminnolla. Tarvittaessa kuitenkin kerrataan vaadittavat toiminnot oppilaiden kanssa ennen ohjelmointiosuuden alkamista.

Oppituntien kulku:
Aloitetaan oppitunti klassisella todennäköisyydellä käymällä läpi käsitteet alkeistapaus, suotuisat alkeistapaukset ja todennäköisyyden käsite. Lisäksi opetellaan laskemaan jonkin tapahtuman A todennäköisyys P(A). Todennäköisyyden laskemista harjoitellaan alkuun kirjallisten tehtävien avulla, jotta aihe tulee kaikille tutuksi ja jokaisella oppilaalla on jonkinlainen käsitys siitä, miten määritetään suotuisat alkeistapaukset ja miten ilmiön todennäköisyys lasketaan.

Tämän jälkeen siirrytään tilastolliseen todennäköisyyteen, jossa tutkitaan tapahtumia kokeellisesti ja tilastoimalla saatuja tuloksia. Nämä todennäköisyydet ovat suuntaa antavia keskimääräisiä lukuja. Mitä suurempi satunnaisesti valittujen tutkimuskohteiden tai kokeen toistojen määrä on, sitä luotettavampaa tietoa saadaan. Tapahtuman A tilastollinen todennäköisyys saadaan jakamalla tapahtuman A esiintymismäärä kaikkien tapausten määrällä.

Kruuna vai klaava? -tehtävä
Oppilaat saavat kolikon ja tehtävän määrittää todennäköisyys sille, että yhdellä kolikon heitolla saadaan klaava. Tämän jälkeen oppilaat tilastoivat paperille tai excel -tiedostoon a) 10 b) 30 ja c) 50 peräkkäisen kolikonheiton tuloksen ja laskevat tilastolliset

todennäköisyydet kruunalle ja klaavalle. Oppilaiden tulee verrata klassista todennäköisyyttä ja tilastollista todennäköisyyttä keskenään; vastaavatko tulokset toisiaan? Muuttaako heittomäärien lisääminen tulosta?

Tehtävän jälkeen oppilaiden kanssa keskustellaan siitä, oliko heittojen tilastoiminen käsin työlästä ja siirrytään ohjelmointiosuuteen.

Ohjelmointi Pythonilla:

1. Aloitetaan ohjelmoimalla yksinkertainen ohjelma, joka arpoo satunnaisen luvun halutulta väliltä. Funktio randint(a, b) antaa satunnaisen kokonaisluvun väliltä a...b. Esimerkiksi seuraava ohjelma simuloi nopan heittämistä:

```
from random import *

x = randint(1,6)
print(x)
```

Suorita

```
3
```

Jotta saadaan ohjelma, joka arpoo satunnaisen luvun, tulee Pythonin standardikirjastosta hakea moduuli random. Oppilaiden kanssa harjoitellaan käyttämään standardikirjastoa ja opetellaan funktion randint(a, b) toiminta vaihtelemalla otantaväliä. Oppilaat voivat tässä kohtaa myös kerrata, miten lasketaan saadun nopan silmäluvun todennäköisyys.

2. Seuraavaksi oppilaiden tulisi laatia ohjelma, joka simuloi kymmentä nopan heittoa. Kyseinen ohjelma voisi olla esimerkiksi tälläinen:

```
from random import *

for i in range(10):
    x=randint(1,6)
    print(x)
```

Suorita

```
4
2
3
6
6
1
6
2
```

3. Lopuksi toistetaan kruuna vai klaava -tehtävä yllä olevan ohjelman avulla muuttamalla väliksi (1, 2), missä 1 = kruuna ja 2 = klaava sekä käyttämällä
a) range(10) b) range(30) c) range(50) .
Oppilaat laskevat tilastolliset todennäköisyydet saadulle datalle.

4. Ohjelmointia voi jatkaa Tie koodariksi -sivuston luvun 15 tehtävien parissa, joissa harjoitellaan simulaation toistamista ja laaditaan ohjelma, joka kertoo kuinka monta kruunaa ja klaavaa saadaan esimerkiksi sadalla heitolla, jolloin ei tarvitse itse laskea näitä datasta.

Lisätehtävä: Oppilas voi laatia ohjelman lottoarvontaan. Tällöin tulisi laatia ohjelma, joka arpoo satunnaisen luvun väliltä 1-40 ja laajentaa tätä arpomaan seitsemän eri lukua samalta väliltä. Vinkki tähän: https://python-k20.mooc.fi/osa-7/2-satunnaisuus

Yhteenveto:
Keskustellaan oppilaiden kanssa ohjelmoinnin merkityksestä kyseisen aihepiirin kannalta. Mitä hyötyä ohjelmoinnista oli? Nopeuttaako ohjelmointi datan keräämistä? Miten muuten ohjelmointia voitaisiin hyötykäyttää todennäköisyyslaskennassa tai tilastojen laatimisessa?

Oppilaat vastaavat itsearviointiin:

	Kyllä	Osittain	En
Osaan kertoa kaverille, miten määritetään esimerkiksi nopan heitossa suotuisat alkeistapaukset ja kaikki alkeistapaukset?			
Osaan laskea, millä todennäköisyydellä noppaa heittämällä saadaan silmäluku 4?			
Tiedän, miten tilastollisesta datasta lasketaan yksittäisen tapahtuman suotuisat tapaukset?			
Opin tänää jotain uutta ohjelmoinnista?			
Osaan laatia ohjelman, joka arpoo satunnaisen nopan silmäluvun?			

Lähteet:
Tie koodariksi -sivusto: https://tie.koodariksi.fi/alkeet/15
Arto vihavainen - Ohjelmoinnin perusteet Python-kielellä:
https://www.cs.helsinki.fi/group/linkki/materiaali/python-perusteet/materiaali.html#19
Ohjelmoinnin perusteet -kurssi: https://python-k20.mooc.fi/osa-7/2-satunnaisuus
Päivi Kulmala - Pro Gradu: Lukiokurssi matemaattisesta ohjelmoinnista Python-kielellä

Peliohjelmointia Pythonilla, 9. lk

Ari Virtanen, CC BY-SA 4.0

22.3.2022

Peliohjelmointia Pythonilla

Aihepiirin valinta ja rajaus

- Aiheena on Pythonin pelikirjasto-ohjelmistoon pygameen tutustuminen. Oppiaineena on matematiikka, koska hyödynnetään geometriaa. Tämä on jatkoa Pythonin perusasioille.
- Oletus on, että oppilaat ovat perehtyneet tekstipohjaiseen ohjelmointiin Pythonilla, mutta haluaisivat myös tuottaa grafiikkaa ja animaatioita vastaavasti kuin Scratchilla.
- Tärkein oppimistavoite on osoittaa, Pythonilla voidaan tuottaa vastaavia pelejä kuin Scratchilla ja kun asiaan hieman perehtyy, vähänkin mutkikkaamman pelin kohdalla itse asiassa helpommin kuin Scratchilla.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

- Tärkein oppimistavoite on siis saada oppilaat näkemään Pythonin monipuoliset käyttömahdollisuudet. Tavoitteena on myös osoittaa, että esimerkiksi jonkin pelin koodaamisessa myös matematiikan tiedot ja taidot ovat hyödyllisiä.
- Arviointikriteerinä on oikeastaan vain se, että oppilas on valmis yrittämään itsekin tuottamaan tehtävissä annetulla tavalla koodia. Tarvittava matematiikka on selostettu oppimateriaalissa, joten sen kohdalla osaamista ei arvioida ollenkaan. Jos olisi käytettävissä enemmän aikaa, tarvittavista matemaattisista asioista voisi olla omat tehtävänsä, jotka voitaisiin arvioida.
- Oppilas voi arvioida osaamistaan sillä, millaisen pelin hän pystyy lopputuloksena tekemään. Ehkä tärkeämpää kuitenkin on se, että oppilas voi oppituntien jälkeen arvioida, onko hän kiinnostunut tämänkaltaisesta ohjelmoinnista. Jos on, oppitunnit antavat hyvän lähtökohdan jatkaa asiaan perehtymistä harrastuspohjalta.

Työskentelyvälineet ja opetusmenetelmät

- Oletus on, että oppilaat ovat jo käyttäneet Pythonia sopivassa ohjelmointiympäristössä ja myös pygame ohjelmointikirjasto on käytettävissä.
- Oppitunteja varten on laadittu oppimateriaali sisältäen myös koodit. Opettaja voisi tunneilla joko mennä oppimateriaalia systemaattisesti läpi tai ottaa lähtökohdaksi koodit ja selittää koodin yksityiskohtia vapaamuotoisesti tarvittaessa hyödyntäen oppimateriaalia.
- Oppilaat, jotka ovat kiinnostuneet Pythonin peruskäytöstä, lienevät myös kiinnostuneita näkemään, miten grafiikkaa käsitellään Pythonissa. Oppilaille, jotka pitävät lausekielistä ohjelmointia Pythonilla hieman tylsänä ja kaipaavat ehkä aiemmin tutustumansa Scratchia, nämä oppitunnit yrittävät osoittaa, että samoja asioita kuin Scratchilla voi tehdä myös Pythonilla.
- Oppitunneilla on tarkoitus tehdä oppimateriaalin tehtäviä sopiviin ryhmiin jakautuen. Oppilaat voisivat tehdä myös pareittain tehtäviä ja esitellä työtään muille joko valmiina ratkaisuina tai esittämällä kysymyksiä ongelmallisiksi osoittautuneista kohdista.
- Periaatteessa oppimateriaalia voi lukea ja tehtäviä tehdä, vaikkei kovasti ohjelmointiosaamista olisikaan. Koodin ymmärtäminen ja erityisesti oppimateriaalin ulkopuolelle menevä koodin oma täydentäminen edellyttävät kuitenkin jo kohtuullista ohjelmointiosaamista (vastaavan tapaista kuin mitä Tie koodariksi opettaa Pythonista).

- Oppikokonaisuuden jälkeen peliohjelmoinnista kiinnostunut oppilas saattaisi olla kiinnostunut erityisesti sellaisesta matematiikan oppituntien sisällöstä, joka olisi sovellettavissa tällaiseen ohjelmointiin.

Ajoitus

- Ensimmäisellä oppitunnilla käsitellään oppimateriaalista Johdanto ja Pelialue käyttöön ja tehdään tehtävät 1 – 3. Jos mahdollista, ensimmäinen oppitunti voisi olla kuitenkin jaettu osiin niin, että vaikka ohjelmointia opettavan oppitunnin lopussa käsiteltäisiin tämä osuus tehtäviä lukuun ottamatta ja oppilaille annettaisiin mahdollisuus kokeilla tehdä tehtäviä itsenäisesti ennen jatkoa. Jos tämä ei ole mahdollista, niin oppitunnilla yritetään jonkin aikaa tehdä tehtäviä ryhmätyönä ja sitten jatketaan oppimateriaalin lukuun Pelaajat kentälle (jossa esitellään tehtävien ratkaisut). Tehtävät 5 – 6 annettaisiin kotitehtäviksi (vapaaehtoisiksi tai pakollisiksi.
- Toisella oppitunnilla käsitellään luku Peli käyntiin, joka sisältää tehtävien 5 – 6 ratkaisut. Jos aikaa riittää, aletaan tekemään myös ryhmätyönä tehtäviä 7 – 8. Tarkoitus olisi, että asiasta kiinnostuneet oppilaat jatkaisivat tehtävien tekemistä oppitunnin ulkopuolellakin ja heille annettaisiin tavalla tai toisella mahdollisuus myöhemmin esitellä koodaamansa peli.

Peliohjelmointia Pythonilla

Ari Virtanen 22.3.2022

Johdanto

Pythonissa on käytettävissä useita ohjelmakirjastoja erilaisiin käyttötarkoituksiin täydentämään Pythonin peruskomentoja.

math-kirjastossa on määritelty matemaattisia funktioita. Tämän kirjaston saa käyttöön kirjoittamalla Python ohjelman alkuun import math. Tämän jälkeen esimerkiksi luvun 5 neliöjuuren voi laskea Python koodissa komennolla math.sqrt(5). Jos ei tarvita math-kirjastosta muita funktioita, voi myös menetellä niin, että ohjelman alkuun kirjoittaa from math import sqrt, ja koodissa neliöjuuren voi laskea sitten tällä tuodulla sqrt-funktiolla tyyliin sqrt(5).

random-kirjastossa on määritelty erilaisia satunnaislukuja tuottavia funktioita. Jos esimerkiksi haluaa simuloida koodissa nopanheittoa, jossa saadaan jokin silmäluvuista 1, 2, 3, 4, 5, 6, tämä onnistuu komennoilla

```
import random
noppa = random.randint(1, 6)
```

Muitakin Pythonin ohjelmakirjastoja on listattu MAOLin sivulla https://maol.fi/materiaalit/kpm/7-luokka/p1/yleist%c3%a4/1-15-kirjastot/

Tällä kahden oppitunnin mittaisella kurssilla tutustutaan erityisesti pygame-kirjastoon. Pygame on grafiikan tuottamiseen ja peliohjelmointiin tarkoitettu Pythonin kirjasto. Asennusohje ja tätä oppimateriaalia perusteellisempi pygamen esittely löytyy mm. Helsingin yliopiston ohjelmoinnin kurssien oppimateriaalista https://ohjelmointi-22.mooc.fi/ (nämä kurssit ovat ilmaisia ja kaikille avoimia). Pygamea käsitellään tässä materiaalissa osasta 13 eteenpäin https://ohjelmointi-22.mooc.fi/osa-13/1-pygame-kayttoon.

Näillä kahdella oppitunnilla ei ole niinkään tarkoitus oppia tekemään itsenäisesti pygamella pelejä, vaan näyttää, millaisia mahdollisuuksia on käytettävissä. Ohjelmointia oppii parhaiten koodaamalla ja kokeilemalla erilaisia komentoja. Näillä oppitunneilla annetaan paljon valmista koodia, mutta osallistujilla on myös mahdollisuus itse tehdä täydennyksiä ja muutoksia koodiin. Tärkeintä tässä on yrittäminen, sillä se opettaa parhaiten pygamen käyttöä. Vaikka oma koodaus ei tuottaisikaan aivan haluttua lopputulosta, se ei haittaa, koska tunneilla annetaan vaiheittain myös malliratkaisut. Tavoitteena on kuitenkin se, että oppituntien jälkeen oppilas pystyy tekemään malliratkaisuja laajempia koodeja.

Lukijan oletetaan olevan perehtyneen Pythonin peruskomentoihin. Myös Scratchiin tutustumisesta voi olla hyötyä, koska tarkoitus on tehdä samanlaisia graafisia pelejä kuin mitä Scratchillakin. Mitenkään välttämätöntä ei kuitenkaan Scratchin tunteminen ole.

Tarvittaessa kertaa Pythonista seuraavat asiat:

- Kommenttien merkitseminen risuaidalla tai kolmoislainausmerkkien sisälle.
- Vakiot ja muuttujat
- while ja for -silmukat
- if else -lause

- loogiset ja aritmeettiset operaattorit
- aliohjelmat ja parametrit

Pelialue käyttöön

Oppituntien tavoitteena on laatia peli, jossa pelialueella on kaksi maalia ja kaksi pelaajaa, sekä pallo, jonka pelaajat yrittävät saada toisen pelaajan maaliin. Oppituntien aikana tehtävä peli on sellainen, että sitä voi pelata yksinkin.

Ensimmäiseksi otetaan pygame käyttöön ja kokeillaan erilaisten pelialueiden muodostamista. Opettavaisinta olisi itse etsiä ja kokeilla sopivia komentoja alusta lähtien, mutta aikataulupulan vuoksi tunnilla jaetaan puolivalmis koodi koodi1.py, johon voi muutamia kokeiluja tehdä.

Tässä on selitetty koodissa olevat komennot ja lopuksi annetaan tehtävät, joita voi ominpäin yrittää tehdä.

Pygame otetaan käyttöön komennolla

```
import pygame
```

Pelissä esiintyy erilaisia värejä. Värit voidaan Pythonissa määritellä mm. RGB-värimallilla, jossa annetaan lukukolmikko (red, green, blue), jossa luvut ovat väliltä 0 – 255. Ensimmäinen komponetti red kertoo, kuinka paljon värissä on punaista. 0 tarkoittaa, että ei yhtään, 255 tarkoittaa 'täyttä' punaisuutta. Vastaavasti toinen komponentti green kertoo vihreän värin ja kolmas blue sinisen värin määrän. Peliä varten on määritelty vakioiksi (tässä esityksessä vakiot kirjoitetaan aina isoilla kirjaimilla; tarkoitus on, että niiden arvoja ei muuteta sen jälkeen, kun ne on sopivasti määritelty).

```
VALKOINEN = (255, 255, 255)
PUNAINEN = (255, 0, 0)
VIHREA = (0, 255, 0)
SININEN = (0, 0, 255)
MUSTA = (0,0,0)
HARMAA = (128,128,128)
```

Muitakin värejä voi tietenkin määritellä ja nimetä.

Seuraavaksi määritellään varsinaisen pelialueen leveys ja korkeus. Vaikka nämä ajatellaankin vakioiksi, tarkoitus on kokeilemalla etsiä niiden sopivat arvot. Pelialueen yläpuolelle tulee otsikko ja alapuolelle tekstikenttä, jonka korkeus myös määritellään:

```
LEVEYS = 650
KORKEUS = 100
OTSIKKO = 'Ensimmäinen Pygame kokeiluni'
TEKSTIKENTAN_KORKEUS = 33
```

Pelialue (tässä otettu sille käyttöön nimitys ’ALUE’), pelialueen väri (tässä valittu harmaa) ja otsikko määritellään seuraavasti:

```
ALUE = pygame.display.set_mode((LEVEYS, KORKEUS + TEKSTIKENTAN_KORKEUS))
ALUE.fill(HARMAA)
```

```
pygame.display.set_caption(OTSIKKO)
```

Näitä komentoja ei ole syytä opetella ulkoa, vaan ne kannattaa katsoa ja tarvittaessa kopioida mallikoodista tiedostosta koodi1.py. Jos kuitenkin kopioit Python käskyjä tästä lukemastasi tekstistä, huolehdi siitä, että sisennykset tulevat oikein.

Pelialueen alapuolella on varattu tilaa tekstille (esimerkiksi pelitilanteen kirjoittamista varten) ja tarkoitus ei ole pelata tällä alueella. Erotetaankin se varsinaisesta pelialueesta esimerkiksi punaisella viivalla, jonka paksuus on 3:

```
pygame.draw.line(ALUE, PUNAINEN, (0, KORKEUS), (LEVEYS, KORKEUS), 3)
```

Viiva piirretään siis komennolla pygame.draw.line(<parametrit>). Parametreista ensimmäinen viittaa pelialueelle annettuun nimeen. Sitten tulee viivan väri ja viivan päätepisteet xy-koordinaatistossa. Piste (0, KORKEUS) vastaa aivan vasemmalla pelialueen ja tekstikentän välissä olevaa pistettä. Piirrettävän viivan paksuus on viimeinen parametri.

Pygame-kirjaston käyttö aloitetaan seuraavalla funktiokutsulla:

```
pygame.init()
```

Yllä olevilla komennoilla saadaan jo näkymään välähdys pelikentästä, mutta ei muuta.

Minimikoodi pelialueen näyttämiseksi olisi seuraavalla

```
while True:
        pygame.display.update()
```

Tässä on kuitenkin ikuinen luuppi. Pelin olisi syytä olla myös suljettavissa. Alla olevassa koodin pätkässä tämä on toteutettu käyttämällä muuttujaa peliJatkuu, joka on aluksi tosi ja muuttuu sitten epätodeksi (jolloin peli päättyy), kun käyttäjä klikkaa oikealla yläkulmassa olevaa rastia X. Koodissa on myös kirjoitettu tämä ohje tekstikenttään. Ohjeen fontiksi on valittu ’arial’, kooksi 30 ja nimeksi yksinkertaisesti ’fontti’:

```
fontti = pygame.font.SysFont('arial', 30)
```

Kirjoittamista varten teksti ensin renderöidään (eli muokataan tulostukselle sopivaan muotoon)

```
tekstiR = fontti.render('Sulje yläkulmassa olevaa rastia X klikkaamalla', True, PUNAINEN)
```

Väriksi on siis valittu punainen. Totuusarvon True merkityksen voi selvittää kokeilemalla muuttaa sen epätodeksi False.

Näiden valmisteluiden jälkeen teksti kirjoitetaan

```
ALUE.blit(tekstiR, (10, KORKEUS + 5))
```

Ensimmäinen parametri on siis renderöity teksti, toinen tekstin alkamiskohdan (x,y)-koordinaatti.

Pelikentän teksteineen piirtäminen näytölle tapahtuu funktiolla

```
pygame.display.update()
```

Muitakin tapoja on, mutta tämä on luonteva, koska jatkossa näytön pelitapahtumia myös päivitetään tällä.

Yhteenvetona koodin loppuosa on siis

```
peliJatkuu = True
while peliJatkuu:
        fontti = pygame.font.SysFont('arial', 30)
        tekstiR = fontti.render(
        'Sulje yläkulmassa olevaa rastia X klikkaamalla', True, PUNAINEN)
        ALUE.blit(tekstiR, (10, KORKEUS + 5))
        pygame.display.update() # Piirretään pelialue
        for tapahtuma in pygame.event.get():
                if tapahtuma.type == pygame.QUIT:
                #Rastia klikattu
                peliJatkuu = False
                #Peli päättyy
```

Tässä näppäinten ja hiirten painalluksia ja liikkeitä kerätään funktiolla pygame.event.get(). For-silmukassa käydään näitä tapahtumia läpi, silmukan muuttujan nimeksi on valittu kuvaava ’tapahtuma’. Tässä vaiheessa ainut huomioitava tapahtuma on kuitenkin vain ylänurkan rastin klikkaaminen, joka lopettaa pelin.

Tehtävät.

Tehtävä 1. Suorita Pythonilla oppitunnilla annettu koodi koodi1.py. Jos et näe näytöllä pelialuetta, tarkista, että se ei ole piilossa jonkun näytöllä auki olevan ikkunan takana. Jos haluat käyttää pygamea omassa koneessasi, asennusohjeet löytyy mm. tekstistä mainitusta Helsingin yliopiston materiaalista tai MAOLin sivulta
https://maol.fi/materiaalit/taidetta-ohjelmoimalla/1-johdanto/python-ohjelmointi-pygame/

Tehtävä 2. Kokeile muodostaa erilaisia pelialueita. Vaihtele värejä, kokoja, otsikon ja pelialueen alaosan tekstejä. Voit myös määritellä omia värejä.

Tehtävä 3. Nyt punainen viiva erottaa pelialueen ja sen alapuolella olevan tekstikentän. Ympäröi koko pelialue punaisilla viivoilla lisäämällä sellainen vasempaan ja oikeaan reunaan ja yläosaan. Toteuta tämä niin, että viivat tulevat oikeisiin kohtiin, vaikka pelialueen kokoa vaihdellaankin eli hyödynnä vakioita LEVEYS ja KORKEUS (joiden arvoja voidaan määrittelyvaiheessa muuttaa).

Pelaajat kentälle

Jaettavassa koodissa koodi2.py on pelialueelle on sijoitettu yksi ympyrä, jolle käytetään nimitystä 'maalivahti' ja jota voi siirrellä hiirellä, ja toinen ympyrä, nimityksenä 'pelaaja', jota voi siirrellä nuolinäppäimillä. Tehtävänä on sijoittaa pelialueelle myös kaksi suorakulmioita, joille käytetään nimityksiä 'maali1' ja 'maali2' ja muita pienempi ympyrä, nimityksenä 'pallo'.

`import pygame` komennon lisäksi alussa on hiiren kursorin ohjelmallista siirtämistä varten `from pygame import mouse` ja suorakulmioiden piirtämistä varten `from pygame.locals import Rect`

Alkuosa koodista on tuttua edellisestä koodista, tehtävänä ollut pelialueen ympäröinti punaisilla viivoilla on toteutettu aliohjelmalla `def pelialueen_reunat`.

Ympyröiden piirtäminen on toteutettu olio-ohjelmointitekniikalla muodostamalla yleinen luokka Ympyra. Olio-ohjelmointitekniikkaa ei itse tarvitse näillä oppitunneilla käyttää. Luokan Ympyra voi ajatella niin, että siinä on kerätty yhteen ympyrän piirtämiseen Pygamella tarvittavat tiedot.

Mitä tietoja ympyrän piirtämiseksi sitten tarvitaan? Esimerkiksi väri, ympyrän keskipiste ja säde riittävät. Kun nämä ominaisuudet on tiedossa, ympyrä voidaan piirtää funktiolla `pygame.draw.circle`. Tätä funktion nimeä ja sen parametreja ei tarvitse kuitenkaan muistaa, vaan piirtäminen tapahtuu hyödyntämällä luokan Ympyra määrittelyssä olevaa piirra-metodia.

Luokan (class) Ympyra määritelmä on seuraava:

```
#'Ympyra' niminen luokka
class Ympyra:
	def __init__(self, vari, keskipiste, sade):
		self.vari = vari
		self.keskipiste = keskipiste
		self.sade = sade
	def piirra(self):
		pygame.draw.circle(ALUE,self.vari, self.keskipiste, self.sade, width = 0)
```

’self’ viittaa luokan Ympyra olioon. Sillä on seuraavanlaisia ominaisuuksia: ympyrän väri self.vari, ympyrän keskipiste koordinaatiossa self.keskipiste ja ympyrän säde self.sade.

def __init__(self, vari, keskipiste, sade) on luokan Ympyra ns. konstruktio. Sitä käytetään määriteltäessä luokkaan Ympyra kuuluvia olioita. Jos halutaan esimerkiksi määritellä ympyrä, jolle käytetään nimitystä ’pallo’, jonka väri on musta, keskipiste on pisteessä (100,100) ja säde 10, niin se tehdään seuraavasti:

```
pallo = Ympyra(MUSTA, (100,100), 10)
```

Kun on määritelty esimerkiksi pallo niminen luokan Ympyra olio, se piirretään luokan piirra-metodilla. Piirra-metodi ei vielä sinänsä piirrä ympyrää näkyviin pelialueelle ALUE, vaan koodissa täytyy esiintyä myös näytön päivitys funktiolla 'pygame.display.update()'.

Koodissa on määritelty seuraavat ympyrät:

```
REUNASTA = 120
maalivahti = Ympyra(MUSTA, (REUNASTA, KORKEUS/2), 35)
pelaaja = Ympyra(MUSTA, (LEVEYS - REUNASTA, KORKEUS/2), 35)
```

Kummankin väri on musta ja säde 35. Ympyröiden sijoittamisessa on käytetty vakiota REUNASTA, jonka arvoksi on asetettu etäisyys reunasta. ’maalivahtia’ esittävän ympyrän keskipiste on vasemmasta reunasta tällä etäisyydellä, ’pelaaja’ taasen oikeasta reunasta. Kummankin keskipiste sijaitsee pelialueen keskellä korkeuden suhteen.

’maalivahtia’ voi liikuttaa hiirellä, ’pelaajaa’ nuolinäppäimillä. Tämä on toteutettu pelin while-silmukassa. Ennen sen koodiin tutustumista perehdytään kuitenkin luokan Suorakulmio määrittelyyn, jonka avulla voi kentälle piirtää suorakulmiot esittämään maaleja.

```
class Suorakulmio:
        def __init__(self, vari, x_sijainti, y_sijainti, leveys, korkeus):
                self.vari = vari
                self.x = x_sijainti
                self.y = y_sijainti
                self.l = leveys
                self.k = korkeus

        def piirra(self):
                rect = Rect(self.x, self.y, self.l, self.k)
                pygame.draw.rect(ALUE, self.vari, rect, width = 0)
```

Luokan konstruktorissa def __init__(self, vari, x_sijainti, y_sijainti, leveys, korkeus) annetaan suorakulmion väri, ja sen vasemman ylänurkan sijaintipaikka (x_sijainti, y_sijainti) koordinaatissa. Lisäksi tarvitaan suorakulmion leveys ja korkeus. Nämä sijaintitiedot muodostavat attribuutit self.x, self.y, self.l ja self.k. Esimerkiksi suorakulmion oikea alanurkka sijaitsee pisteessä (self.x + self.l, self.y + self.k). Luokan metodissa piirra() on suorakulmion piirtämiseen Pygamessa tarvittavat tiedot.

Tarkastellaan seuraavaksi, miten luokkaan Ympyra kuuluvaa ’maalivahtia’ voidaan liikuttaa hiirellä. Kun tapahtumia (näppäinten painalluksia ja hiiren liikuttelua) käydään läpi silmukassa
`for tapahtuma in pygame.event.get():`
niin hiiren liikuttaminen tunnistetaan if-lauseessa
`if tapahtuma.type == pygame.MOUSEMOTION:`

’maalivahdin’ liikuttaminen hiirellä tapahtuu nyt yksinkertaisesti muuttamalla sen keskipisteen koordinaateiksi kursorin sijainti:

```
uusi_keskipiste = (tapahtuma.pos[0], tapahtuma.pos[1])
maalivahti.keskipiste = uusi_keskipiste
```

ja sitten sopivassa paikkaa käyttämällä piirra-metodia `maalivahti.piirra()` ja näytön päivitystä `pygame.display.update()`

’pelaajaa’ liikutellaan nuolinäppäimillä. Periaatteessa tämänkin voi toteuttaa yksinkertaisesti niin, että nuolinäppäimen painallus muuttaa ’pelaajan’ keskipisteen sijaintia. Ensiksi tunnistetaan tapahtuma, että jotain näppäintä yleensä painetaan, if-lauseessa

```
if tapahtuma.type == pygame.KEYDOWN:
```

Sen jälkeen tutkitaan if-lauseessa, mitä näppäintä on painettu

```
if tapahtuma.key == pygame.K_<tunnus>
```

jossa <tunnus> on Pygamessa käytettävä näppäimen tunnus. Koodissa esiintyy seuraavat

- pygame.K_UP nuolinäppäin ylös
- pygame.K_DOWN nuolinäppäin alas
- pygame.K_LEFT nuolinäppäin vasemmalle
- pygame.K_RIGHT nuolinäppäin oikealle
- pygame.K_SPACE välilyönti

Lisää Pygamen näppäinten koodeja löytyy netistä https://www.pygame.org/docs/ref/key.html#key-constants-label.

Jos haluttaisiin siirtää ’pelaajaa’ yksi askel ylös näppäimellä ’nuoli ylös’, se voisi tapahtua määrittelemällä

```
if tapahtuma.key == pygame.K_UP:
    pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] + 1)
```

Mutta tällöin nuolinäppäintä täytyy painaa aina uudestaan, jotta saisi yhden askeleen siirtymisen tapahtuvan. Kun tätä kokeilee, huomaa liikuttamisen tuntuvan kömpelöltä. Koodissa onkin liikuttaminen nuolinäppäimillä toteutettu yleisesti käytettävällä jatkuvalla tavalla. Idea on, että ’pelaaja’ liikkuu esimerkiksi ylöspäin niin kauan kuin näppäin ’nuoli ylös’ on painettuna alas ja liike pysähtyy, kun nuolinäppäin vapautetaan. Liikuttamista ohjataan nyt muuttujien avulla, esimerkiksi kun muuttujalla ’ylös’ on arvo True, niin ’pelaaja’ liikkuu ylöspäin, mutta ei liiku ylöspäin, kun muuttujalla on arvo False. Muuttujalle ’ylös’ asetetaan arvo True painamalla nuolinäppäintä ’nuoli ylös’

```
if tapahtuma.key == pygame.K_UP:
    ylos = True
```

Muuttujan arvon asettaminen epätodeksi tapahtuu if-lauseella

```
if tapahtuma.type == pygame.KEYUP:
    ylos = False
```

Tässä siis ehtolauseessa pygame.KEYUP vastaa näppäimen vapauttamista. Lisäksi on muuttujan ’ylos’ alkuarvoksi asetettava False ennen while-silmukkaa. ’Pelaajan’ keskipistettä y-akselin suhteen kasvatetaan yhdellä aikayksikköä kohti if-lauseella

```
if ylos:
    pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] - 1)
```

Jos liikkuminen tuntuu hitaalta, sitä voi nopeuttaa kasvattamalla vähentäjää. Vastaavasti voidaan käsitellä muutkin liikkumisen suunnat.

Luonnollisesti pelaajan liikkuminen näytöllä edellyttää myös näytön päivittämistä. Siinä ensin poistetaan kaikki oliot pelialueelta funktiolla

```
ALUE.fill(VARI)
```

jossa VARI on siis pelikentän taustaväri. (Voit kokeilla, mitä tapahtuu, jos tätä funktiota ei ole.) Tämän jälkeen *kaikki* pelikentän oliot on piirrettävä uudelleen, esim. ’pelaaja’ metodilla

```
pelaaja.piirra()
```

ja näyttö päivitettävä

```
pygame.display.update()
```

Edellä sanottiin ' kasvatetaan yhdellä aikayksikköä kohti'. Mikä tämä 'aikayksikkö' sitten on? Se riippuu esimerkiksi käytettävästä tietokoneesta. Jotta peli toimisi samalla tavalla eri koneissa, voidaan ottaa käyttöön 'kello'. Käyttöönotto tapahtuu funktiolla

```
kello = pygame.time.Clock()
```

Kellon 'tikitys' tahdistaa nyt pelitapahtumia. 'Tikitystä' säädelleen taas seuraavasti:

```
kello.tick(NOPEUS)
```

jossa vakiolla NOPEUS on sopivaksi katsottu arvo (sopivaa arvoa voi etsiä aloittamalla arvosta 60). Mitä isompi tuo luku on, sitä nopeampaa ovat sellaiset pelitapahtumat kuin ’pelaajan’ liikkuminen nuolinäppäimen ollessa alhaalla.

Pelitapahtumia säätelevässä while-silmukassa on myös if-lause

```
if tapahtuma.key == pygame.K_SPACE:
                maalivahti.keskipiste = (REUNASTA, KORKEUS/2)
                pelaaja.keskipiste = (LEVEYS - REUNASTA, KORKEUS/2)
                mouse.set_pos(maalivahti.keskipiste[0],maalivahti.keskipiste[1])
```

Se ennakoi jatkoa, jossa halutaan tietyissä tilanteissa palata pelialueella alkuasetelmaan. Nyt kun painetaan välilyöntiä, yllä olevan seurauksena ’maalivahti’ ja ’pelaaja’ siirtyvät pelin alussa oleviin paikkoihin ja `mouse.set_pos` siirtää hiiren kursorin osoittamaan ’maalivahtia’ tässä alkuperäisessä paikassa.

Koodi kokonaisuudessaan on tiedostossa koodi2.py.

Tehtävät

Tehtävä 4. Määrittele 'maalivahtia' ja 'pelaajaa' pienempi ympyrä, anna sille nimeksi 'pallo' ja yritä sijoittaa se ’pelaajan’ vasemmalle puolelle (siis ’pelaajan’ eteen oikeasta reunasta katsoen) parhaassa tapauksessa niin, että 'pallo' ja 'pelaaja' koskettavat toisiaan alkutilanteessa. 'Pallon' ei tarvitse kuitenkaan seurata 'pelaajan' liikkeitä.

Tehtävä 5. Lisää vasempaan reunaan kiinni, korkeuden suhteen keskelle sopivan kokoinen suorakulmio, jonka voit käyttää nimeä 'maali1', ja vastaavasti 'maali2' oikeaan reunaan. Valitse koot niin, että maalit eivät kosketa 'maalivahtia' ja 'pelaajaa'.

Tehtävä 6. 'Pelaajaa' voi liikutella nyt nuolinäppäimillä vain ylös ja alas. Lisää liikkumiset vasemmalle ja oikealle.

EXTRATEHTÄVÄ. Nuolinäppäimillä 'pelaaja' saadaan pelialueen ulkopuolellekin. Miten tämä voitaisiin estää? 'Maalivahti' voi käydä kentän missä kohtaa vain. Miten liikkumista voisi rajoittaa?

Peli käyntiin

Vastaava koodi koodi3.py

Koodiin on nyt tehty paljon lisäyksiä, joilla saadaan aikaan ensimmäinen varsinainen pelikokemus. Aika ei riitä koodin yksityiskohtaiseen läpikäyntiin, vaan nyt käsitellään vain pääpiirteitä, ja jätetään oman harrastuksen varaan tarkempi perehtyminen koodiin. Tällöin jos ei ymmärrä jotain koodin kohtaa, kannattaa pienillä muutoksilla kokeilla, mikä merkitys tietyllä koodin osalla on. Toisaalta vaikka ei koodiin ehtisikään perinpohjaisesti perehtymään, lienee kuitenkin mahdollista, jos vain innostusta riittää, tehdä lopussa olevat ylimääräiset tehtävät ja saada aikaiseksi jo kunnollinen peli.

Koodissa on ratkaistu edellisen kerran tehtävät ja piirretty pelialueelle ’maalivahti’, ’pelaaja’, ’pallo’ ja kaksi ’maalia’. ’Pallo’ liikkuu itsestään pelialueella nopeuden vaihdellessa. Jos ’pallo’ osuus vasempaan ’maaliin’, pelitilannetta päivitetään tämän mukaisesti. ’Maalivahti’ pystyy torjumaan ’pallon’. Tässä versiossa ei kuitenkaan pallon osumista oikeanpuolen ’maaliin’ pidetä kirjaa, eikä ’pelaaja’ pysty vielä vaikuttamaan ’pallon kulkuun.

Pelitilanteen alkuasetelma saadaan aikaiseksi aliohjelmalla `def alkuasetelma`. Se suoritetaan alussa, mutta myös kun 'pallo' menee 'maaliin' tai reunan yli.

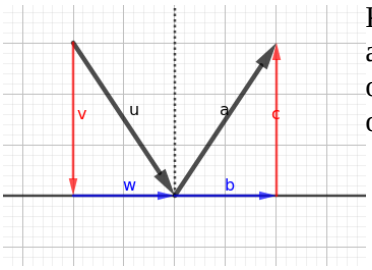
Tarkastellaan pallon liikkumista pelialueella hieman yksityiskohtaisemmin. Luokan Ympyra määrittelyyn on lisätty attribuutti self.suunta, jotta ympyrä olisi mahdollista saada liikkumaan itsestään. Sen suunta on lukupari, tai oikeammin vektori (a,b), jossa a kertoo paikan vaihdon x-akselin suhteen aikayksikössä, b y-akselin suhteen. Sen alkuarvo on (0,0), joka vastaa liikkumattomuutta. Metodilla `muuta_suuntaa(self, x,y)` voidaan suunnaksi vaihtaa vektori (x,y).

Itsestään liikkuva ’pallo’ katoaisi nopeasti pelialueelta. Koodissa onkin asetettu pallo kimpoamaan reunoista. Fysikaalisessa maailmassa tällainen kimpoaminen riippuu monesta asiasta (ks. esim. https://fi.wikipedia.org/wiki/Kimmoisuusaste), mutta tällaisessa pelissä, jossa pallon nopeuttakaan ei muuteta törmäyksen jälkeen, on luontevaa ajatella pallon käyttäytyvän samalla tavalla kuin valo

heijastuu peilistä: tulokulma reunan normaalin kanssa on sama kuin heijastuskulma. Esimerkiksi Scratchissa esineen kimpoaminen reunasta on toteutettu tämän periaatteen mukaisesti.

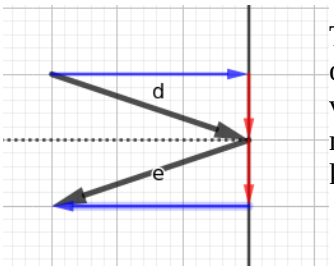
Metodissa reuna_kosketus(self) tarkkaillaan reunakosketuksia. ’Pallo’ koskettaa esimerkiksi vasenta reunaa, jos ’pallon’ keskipiste on tasan ’pallon’ säteen verran vasemmasta reunasta eli
self.keskipiste[0] - self.sade == 0
Peliohjelmassa tulee kuitenkin helposti tilanteita, jossa ’pallo’ ehtii liukua jo reunan ylikin, joten tästä syystä ehdoksi kannattaa tässä vasemman reuna tapauksessa asettaa
self.keskipiste[0] - self.sade <= 0
Vastaavasti tarvittavin muutoksin tutkitaan muutkin reunakosketukset.

Kun ’pallo’ sitten osuu reunaan, sen reunasta kimpoamisen toteuttaminen edellä esitetyn periaatteen mukaisesti riippuu siitä, onko kyseessä pystysuuntainen vai vaakasuuntainen reuna. Jos reuna on vaakasuuntainen, niin tilanne on seuraavan kaltainen (alla pallo tulee vasemmalta, sen suunta on vektorin u mukainen, se osuu alareunaan ja uusi suunta on vektorin a mukainen):



Kun tulosuunta u ja menosuunta a jaetaan komponentteihin x- ja y-akseleiden suhteen, niin x-akselin suuntaisten komponentit w ja b ovat samat vektorit, mutta y-akselin suuntaiset komponentit v ja c ovat toistensa vastavektoreita eli c = -v.

Tilanne on sama, tuli pallo sitten vasemmalta tai oikealta ja osuiko ala- tai yläreunaan (kokeile piirtämällä). Mutta jos pallo tulee ylhäältä tai alhaalta ja osuu vasempaan tai oikeaan reunaan, tilanne on seuraavan kaltainen:



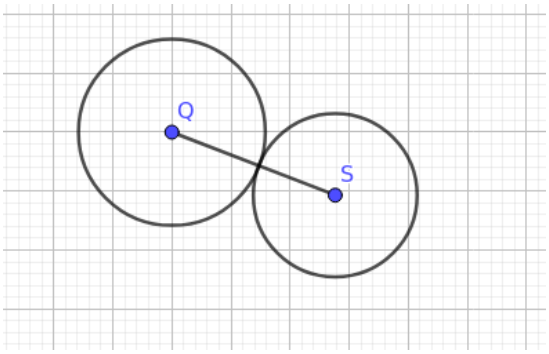
Tulosuunnan d ja menosuunnan e komponentit y-akselin suhteen ovat samat, mutta nyt x-akselin suuntaiset komponentit ovat vastakkaiset. Kuviossa pallo tulee ylhäältä ja osuu oikeaan reunaan, mutta muutkin kimpoamiset vasemmasta tai oikeasta reunasta käyttäytyvät samanlaisesti.

Metodi def kimpoaa(self) on koodattu tämän periaatteen mukaisesti.

Aliohjelmassa def maalivahti_pallo(maalivahti, pelaaja, pallo) käsitellään, mitä tapahtuu, kun 'maalivahti' ja 'pallo' koskettavat toisiaan. Ensinnä on tarkasteltu sitä, milloin 'maalivahtia' ja 'palloa' kuvaavat ympyrät koskettavat toisiaan if-lauseessa

```
if (a-c)**2 + (b-d)**2 <= (pallo.sade + maalivahti.sade)**2:
```

jota ennen on määritelty muuttujat a, b, c ja d niin, että (a,b) on 'maalivahdin' ja (c,d) 'pallon' keskipiste. Kaksi ympyrää sivuavat toisiaan, jos niiden keskipisteiden välinen etäisyys on sama kuin niiden säteiden summa:



Q = (a,b)
S = (c,d)
isomman ympyrän säde r_1,
pienemmän r_2.

Kun keskipisteiden välinen etäisyys on pienempi kuin säteiden summa, ympyrät ovat ainakin osittain sisäkkäin. Tällä perusteella saadaan, että ympyrät koskettavat tosiaan, kun

$$\sqrt{(a-c)^2+(b-d)^2} \leqslant r_1+r_2$$

Jotta ei tarvittaisi neliöjuurta math.sqrt, tässä voidaan vasen ja oikea puoli korottaa potenssiin 2, jolloin saadaan koodissa oleva ehto.

Jos haluttaisiin mallintaa kahden pallon kimpoamista toisistaan luonnollisella tavalla, pitäisi huomioida kummankin suuntavektorit törmäyshetkellä ja pallojen ominaisuudet (erityisesti kokoerot) vaikuttaisivat siihen, mitä törmäyksen jälkeen tapahtuu. Mutta kun puhutaan 'maalivahdin' (tai 'pelaajan') ja 'pallon' kohtaamisesta, voidaan sopia vapaasti, mitä tämän jälkeen tapahtuu. ’Maalivahdin’ ja ’pallon’ koskiessa ’pallo’ lähetetään kohti oikeanpuoleista maalia aliohjelmassa def maalivahti_pallo.Tällöin voi käydä niin, että pallo ehtii menemään reunan yli, jolloin palataan alkuasetelmaan aliohjelmalla def pallo_pelialueen_ulkopuolella.

'Pallon' osumista 'maaliin' käsitellään aliohjelmassa def pallo_maalissa. Nykyisessä koodissa tarkkaillaan vain osumista 'omaan maaliin' eli vasemmalla olevaan suorakulmioon. Aliohjelman koodia voisi täydentää niin, että huomioidaan myös osumat oikeanpuoliseen 'maaliin'.

Maalitilanne ilmoitetaan tekstiruudussa aliohjelman def pelitilanteen_nayttaminen avulla (tällä hetkellä vain 'omaan maaliin' menneet.

Pelialueen piirtäminen tehdään aliohjelmalla def paivita. Tämä aliohjelma piirtää pelialueen ja siellä olevat oliot huomioiden muutokset sijainneissa mukaan lukien pallon mahdollinen kimpoaminen reunasta. Myös pelitilanne kirjataan. Aliohjelmassa tarkistetaan myös, vaikuttaako 'maalivahti' ja 'palloon' (mutta 'pelaajan' ja 'pallon' vuorovaikutusta ei vielä tässä koodissa huomioida) ja onko 'pallo' 'maalissa' (tosin vain 'oma maali' huomioidaan tässä versiossa).

Varsinainen pelin while-silmukka ei oikeastaan rakenteeltaan eroa muuten edellisestä koodista kuin että pelitilanteen päivittäminen hoidetaan aliohjelman def paivita kutsumisella.

Ennen tätä while-silmukkaa suoritetaan aliohjelma def pelin_alkuteksti(). Se ohjeistaa aloittamaan pelin painamalla välilyöntiä. Varsinkin jos kehität peliä monimutkaisemmaksi, tässä kohtaa voisi antaa peliohjeet.

Tehtävät

Tehtävä 7. Huomioi koodissa myös pallon osumat oikeanpuoleiseen maaliin. Päivitä siis aliohjelma ensisijaisesti `def pallo_maalissa`, mutta tee tarvittavat muutokset myös aliohjelmiin `def pelitilanteen_nayttaminen` ja `paivita`.

Tehtävä 8. Muuta koodia niin, että siinä huomioidaan jollain tavalla 'pallon' osuminen 'pelaajaan'. Esimerkiksi 'pallo' voidaan lähettää satunnaiseen suuntaan kohti vasemmanpuoleista 'maalia' sen osuessa 'pelaajaan'.

Extratehtävä. Tee vielä sellaisia muutoksia koodiin, että saadaan yksin tai kaksin pelattava peli, jossa myös pelikokemus on hyvä. Kiinnitä huomiota väreihin, kokosuhteisiin ja pelin nopeuteen.

Yksinkertaisin yksin pelattava peli voisi olla tehtävien 7 ja 8 ratkaisujen jälkeen sellainen, että 'pelaajakin' on paremminkin maalivahti, joka liikkuu automaattisesti oikeanpuoleisen maalin edessä ja torjuu 'pallon' pelkällä tuurilla.

Jos teet kaksin pelattavan pelin, kiinnitä huomiota siihen, että vasemmanpuoleinen pelaaja ei ole hiiren käytön vuoksi ylivoimainen. Yksi ratkaisu olisi tehdä myös siitä näppäimillä ohjattava esim. kirjaimilla ’w’ ylös, ’x’ alas, ’a’ vasemmalle ja ’d’ oikealla. Toinen tapa olisi jotenkin säädellä 'pallon' nopeutta niin, että se liikkuu keskimäärin nopeammin kohti vasenta 'maalia' kuin oikeaa.

Peli

Eräs ratkaisu on tiedostossa koodi4.py

```
#Pelialue käyttöön
#koodi1.py

import pygame

VALKOINEN = (255, 255, 255)
PUNAINEN = (255, 0, 0)
VIHREA = (0, 255, 0)
SININEN = (0, 0, 255)
MUSTA = (0,0,0)
HARMAA = (128,128,128)
#Muitakin värejä voi määritellä


LEVEYS = 650  #Varsinaisen pelialueen leveys,
KORKEUS = 100 #ja korkeus
OTSIKKO = "Ensimmäinen Pygame kokeiluni"
TEKSTIKENTAN_KORKEUS = 33

ALUE = pygame.display.set_mode((LEVEYS, KORKEUS + TEKSTIKENTAN_KORKEUS))
ALUE.fill(HARMAA)
pygame.display.set_caption(OTSIKKO)

#Piirretään punainen viiva
pygame.draw.line(ALUE, PUNAINEN, (0, KORKEUS), (LEVEYS, KORKEUS), 3)

pygame.init()


peliJatkuu = True
while peliJatkuu:
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR = fontti.render("Sulje yläkulmassa olevaa rastia X klikkaamalla",
True, PUNAINEN)
    ALUE.blit(tekstiR, (10, KORKEUS + 5))
    pygame.display.update()  # Piirretään pelialue
    for tapahtuma in pygame.event.get():
        if tapahtuma.type == pygame.QUIT:
            #Rastia klikattu
            peliJatkuu = False
            #Peli päättyy
```

```
#Koodi 2

import pygame
from pygame import mouse
from pygame.locals import Rect

VALKOINEN = (255, 255, 255)
PUNAINEN = (255, 0, 0)
VIHREA = (0, 255, 0)
SININEN = (0, 0, 255)
MUSTA = (0,0,0)
HARMAA = (128,128,128)

LEVEYS = 1000
KORKEUS = 500
OTSIKKO = "Toinen Pygame kokeiluni"
TEKSTIKENTAN_KORKEUS = 33

ALUE = pygame.display.set_mode((LEVEYS, KORKEUS + TEKSTIKENTAN_KORKEUS))
VARI = HARMAA
ALUE.fill(VARI)

#Pelialueen ympäröinti viivoin.
def pelialueen_reunat(viivan_vari, viivan_leveys):
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (LEVEYS, 0), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, KORKEUS), (LEVEYS, KORKEUS),
viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (0, KORKEUS), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (LEVEYS-viivan_leveys, 0),
(LEVEYS-viivan_leveys, KORKEUS), viivan_leveys)

#Punainen viiva, viivan leveys 3
pelialueen_reunat(PUNAINEN, 3)

pygame.display.set_caption(OTSIKKO)
pygame.init()

#'Ympyra' niminen luokka
class Ympyra:
    def __init__(self, vari, keskipiste, sade):
        self.vari = vari
        self.keskipiste = keskipiste
        self.sade = sade

    def piirra(self):
        pygame.draw.circle(ALUE,self.vari, self.keskipiste, self.sade, width  = 0)
        #Koodissa oltava komento 'pygame.display.update()' piirtämistä varten.


REUNASTA = 120 #
maalivahti = Ympyra(MUSTA, (REUNASTA, KORKEUS/2), 35)
#Tätä voi liikutella hiirellä vapaasti
pelaaja = Ympyra(MUSTA, (LEVEYS - REUNASTA, KORKEUS/2), 35)
```

```
#Tätä voi liikutella nuolinäppäimillä.

#'Suorakulmio' niminen luokka
class Suorakulmio:
    def __init__(self, vari, x_sijainti, y_sijainti, leveys, korkeus):
        #Tehdään suorakulmio, jonka vasen ylänurkka on pelialueen
        #koordinaatistossa kohdassa (x_sijainti, y_sijainti) ja joka on kokoa
leveys X korkeus
        self.vari = vari
        self.x = x_sijainti
        self.y = y_sijainti
        self.l = leveys
        self.k = korkeus

    def piirra(self):
        rect = Rect(self.x, self.y, self.l, self.k)
        pygame.draw.rect(ALUE, self.vari, rect, width = 0)

kello = pygame.time.Clock()
NOPEUS = 60 #Tässä voi kokeilla eri nopeuksia
peliJatkuu = True
ylos = False #'pelaaja' ei oletusarvoisesti liiku ylös,
alas = False #eikä alas.
while peliJatkuu:
    kello.tick(NOPEUS)
    ALUE.fill(VARI) #Pelikentän taustaväri
    pygame.draw.line(ALUE, PUNAINEN, (0, KORKEUS), (LEVEYS, KORKEUS), 3)
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR = fontti.render("Tänne voidaan kirjata pelitilanne", True, PUNAINEN)
    ALUE.blit(tekstiR, (10, KORKEUS + 5))
    maalivahti.piirra()
    pelaaja.piirra()
    pygame.display.update() #Piirretään ja päivitetään pelialue
    for tapahtuma in pygame.event.get():
        if tapahtuma.type == pygame.MOUSEMOTION:
                uusi_keskipiste = (tapahtuma.pos[0], tapahtuma.pos[1])
                maalivahti.keskipiste = uusi_keskipiste


        if tapahtuma.type == pygame.KEYDOWN: #jotain näppäintä on painettu

            if tapahtuma.key ==  pygame.K_UP:
                ylos = True

            if tapahtuma.key ==  pygame.K_DOWN: #Jos näppäin oli 'nuoli alas'

                alas = True #liikutaan alas

            #if tapahtuma.key == pygame.K_LEFT:
            #Tässä voi määritellä nuolinpäimen <- painamisesta seuraavan toiminnon

            #if tapahtuma.key == pygame.K_RIGHT:
            #Tässä voi määritellä nuolinpäimen -> painamisesta seuraavan toiminnon
```

```
            if tapahtuma.key == pygame.K_SPACE:
                maalivahti.keskipiste = (REUNASTA, KORKEUS/2)
                pelaaja.keskipiste = (LEVEYS - REUNASTA, KORKEUS/2)
                mouse.set_pos(maalivahti.keskipiste[0], maalivahti.keskipiste[1])

        if tapahtuma.type == pygame.KEYUP: #alaspainettu näppäin vapautetaan
            ylos = False  #Lopetetaan liikkuminen (ylos tai alas)
            alas = False

        if tapahtuma.type == pygame.QUIT:
                peliJatkuu = False

    if ylos:
                pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1]
- 1)

    if alas:
                pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1]
+ 1)
                #Ks. yllä, +1 siis yksi askel, +2 kaksi, jne.
```

```
#Peli käyntiin
#koodi3.py

import pygame
from random import randint #Käytetään satunnaislukuja
from pygame.locals import Rect #Tätä tarvitaan, suorakulmioissa esiintyy 'Rect'
from pygame import mouse

global tauko
#peli jatkuu, kun tämä globaali muuttuja on epätosi

VALKOINEN = (255, 255, 255)
PUNAINEN = (255, 0, 0)
VIHREA = (0, 255, 0)
SININEN = (0, 0, 255)
MUSTA = (0,0,0)
HARMAA = (128,128,128)

LEVEYS = 1000
KORKEUS = 500
REUNASTA = 120 #'maalivahdin' ja 'pelaajan' etäisyys lähimmästä reunasta
alkutilanteessa
OTSIKKO = "Melkein valmis peli"
TEKSTIKENTAN_KORKEUS = 33


class Ympyra:

    def __init__(self, vari, keskipiste, sade):
        self.vari = vari
        self.keskipiste = keskipiste
        self.sade = sade
        self.suunta = (0,0) #Tätä käytetään pallon liikutteluun

    def muuta_suuntaa(self, x,y):
        self.suunta = (x,y)

    def piirra(self):
        global ALUE
        pygame.draw.circle(ALUE,self.vari, self.keskipiste, self.sade, width  = 0)

    def reuna_kosketus(self):
        vasen = (self.keskipiste[0] - self.sade <= 0)
        oikea = (self.keskipiste[0] +  self.sade >= LEVEYS)
        yla = (self.keskipiste[1] - self.sade <= 0)
        ala = (self.keskipiste[1] + self.sade >= KORKEUS)
        return((vasen | oikea), (yla | ala))

    def kimpoaa(self):
        (pystyreuna, vaakareuna) = self.reuna_kosketus()
        if pystyreuna:
            self.muuta_suuntaa(-self.suunta[0], self.suunta[1])
        if vaakareuna:
            self.muuta_suuntaa(self.suunta[0], -self.suunta[1])
```

```
    def siirry(self):
        self.keskipiste = (self.keskipiste[0] + self.suunta[0], self.keskipiste[1]
+ self.suunta[1])


class Suorakulmio:
    def __init__(self, vari, x_sijainti, y_sijainti, leveys, korkeus):
        self.vari = vari
        self.x = x_sijainti
        self.y = y_sijainti
        self.l = leveys
        self.k = korkeus

    def piirra(self):
        global ALUE
        rect = Rect(self.x, self.y, self.l, self.k)
        pygame.draw.rect(ALUE, self.vari, rect, width = 0)

#Määritellään pelin oliot
maalivahti = Ympyra(MUSTA, (0,0), 35)
pelaaja = Ympyra(MUSTA, (0,0), 35)
pallo = Ympyra(MUSTA, (0,0), 10)
maali_vasen = Suorakulmio(MUSTA, 0, 150, 50, 200)
maali_oikea = Suorakulmio(MUSTA, LEVEYS - 50, 150, 50, 200)


#Pelialueen määrittely
ALUE = pygame.display.set_mode((LEVEYS, KORKEUS + TEKSTIKENTAN_KORKEUS))

VARI = HARMAA #Pelikentän taustaväri
ALUE.fill(VARI)

pygame.display.set_caption(OTSIKKO)
pygame.init()

def pelialueen_reunat(viivan_vari, viivan_leveys):
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (LEVEYS, 0), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, KORKEUS), (LEVEYS, KORKEUS),
viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (0, KORKEUS), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (LEVEYS-viivan_leveys, 0),
(LEVEYS-viivan_leveys, KORKEUS), viivan_leveys)

def pelitilanteen_nayttaminen(maalitV, maalitO):
    #maalitO =  osumat oikeaan maaliin
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR = fontti.render(f"Tehdyt maalit: ei vielä laskentaa  Omaan maaliin:
{maalitV}", True, PUNAINEN)
    ALUE.blit(tekstiR, (10, KORKEUS + 5))


def alkuasetelma(maalivahti, pelaaja, pallo, max_x_suunta, max_y_suunta):
    global tauko
```

```
    tauko = True
    maalivahti.keskipiste = (REUNASTA, KORKEUS/2)
    pelaaja.keskipiste = (LEVEYS - REUNASTA, KORKEUS/2)
    pallo.keskipiste = (LEVEYS - REUNASTA - pelaaja.sade, KORKEUS/2)
    mouse.set_pos(maalivahti.keskipiste[0], maalivahti.keskipiste[1])
    #Arvotaan pallolle suunta kohti maalia
    x_suunta = - randint(1,max_x_suunta)
    y_suunta = randint(1,max_y_suunta)
    pallo.muuta_suuntaa(x_suunta, y_suunta)

def pallo_pelialueen_ulkopuolella(pallo):
    x = pallo.keskipiste[0]
    y = pallo.keskipiste[1]
    return (x < 0) | (x > LEVEYS) | (y < 0) | (y > KORKEUS)

def pallo_maalissa(maalivahti, pelaaja, pallo, maali_vasen, maalitV, maali_oikea,
maalit0):
    a = pallo.keskipiste[0]
    b = pallo.keskipiste[1]
    c = maali_vasen.y
    d = maali_vasen.x
    maalissa = (c < b) & (b < c + maali_vasen.k) & (a < d + maali_vasen.l)

    if maalissa:
        maalitV += 1
        ALUE.fill(PUNAINEN) #Välähdys punaista, kun tulee maali
        alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)

    maalit0 = 0 #Tällä hetkellä näistä ei pidetä kirjaa.

    return(maalitV, maalit0)


def maalivahti_pallo(maalivahti, pelaaja, pallo):
    a = maalivahti.keskipiste[0]
    b = maalivahti.keskipiste[1]
    c = pallo.keskipiste[0]
    d = pallo.keskipiste[1]

    if (a-c)**2 + (b-d)**2 <= (pallo.sade + maalivahti.sade)**2:
        #maalivahdilla ja pallolla kosketus
        suunta_x = abs(pallo.suunta[0]) + 1 #itseisarvo, jotta liike kohti oikeaa
reunaa
        if maalivahti.keskipiste[1] < KORKEUS/2:
            suunta_y = 1
        else:
            suunta_y = -1

        pallo.suunta = (suunta_x, suunta_y) #nopeutta voi säädellä kertomalla tai
jakamalla nämä
        if pallo_pelialueen_ulkopuolella(pallo):
            #Tietyissä tilanteissa maalivahti voi työntää pallon pelialueen
ulkopuolelle.
            alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)
```

```
def pelaaja_pallo(maalivahti, pelaaja, pallo):
    #Tässä aliohjelmassa käsiteltään tapausta, jossa 'pallo' osuu 'pelaajaan'.
    pass

def pelin_alkuteksti():
    global tauko
    ALUE.fill(VALKOINEN)
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR1 = fontti.render("Peli alkaa, kun painat välilyöntiä.", True, MUSTA)
    tekstiR2 = fontti.render("Kun pallo menee maaliisi, tulee punainen välähdys.",
True, MUSTA)
    ALUE.blit(tekstiR1, (10, 10))
    ALUE.blit(tekstiR2, (10, 50))
    pygame.display.update()
    tauko = True
    while tauko:
        for tapahtuma in pygame.event.get():
            if tapahtuma.type == pygame.KEYDOWN:
                if tapahtuma.key == pygame.K_SPACE:
                    tauko = False



def paivita(maalitV, maalitO):
    #Tähän kerätään yhteen kaikki pelialueella tehtävät päivitykset
    global tauko
    ALUE.fill(VARI)
    pelialueen_reunat(PUNAINEN, 3)
    pelitilanteen_nayttaminen(maalitV, maalitO)
    pallo.piirra()
    pallo.kimpoaa()
    pallo.siirry()
    maali_vasen.piirra()
    maali_oikea.piirra()
    maalivahti.piirra()
    pelaaja.piirra()
    maalivahti_pallo(maalivahti, pelaaja, pallo)
    maalitV, maalitO = pallo_maalissa(maalivahti, pelaaja, pallo, maali_vasen,
maalitV, maali_oikea, maalitO)
    pygame.display.update()
    if tauko:
        tauko = False
        pygame.time.delay(500)
    return(maalitV,maalitO)

kello = pygame.time.Clock()
NOPEUS = 60 #Tässä voi kokeilla eri nopeuksia
peliJatkuu = True
ylos = False
alas = False
vasemmalle = False
```

```
oikealle = False
torjunnat = 0
maalitV = 0
maalitO = 0
max_x_suunta = 3 #Pallon nopeutta voi säädellä
max_y_suunta = 3 #näillä. Isompi luku johtaa todennäköisesti nopeampaan
liikkumiseen

pelin_alkuteksti()
alkuasetelma(maalivahti, pelaaja, pallo, max_x_suunta, max_y_suunta)
while peliJatkuu:
    kello.tick(NOPEUS)
    maalitV, maalitO = paivita(maalitV, maalitO)

    for tapahtuma in pygame.event.get():
        if tapahtuma.type == pygame.MOUSEMOTION:
                uusi_keskipiste = (tapahtuma.pos[0], tapahtuma.pos[1])

                maalivahti.keskipiste = uusi_keskipiste

        if tapahtuma.type == pygame.KEYDOWN: #jotain näppäintä on painettu

            if tapahtuma.key ==  pygame.K_UP:  #Jos näppäin oli 'nuoli ylös'

                ylos = True  #liikutaan ylos

            if tapahtuma.key ==  pygame.K_DOWN: #Jos näppäin oli 'nuoli alas'

                alas = True #liikutaan alas

            if tapahtuma.key == pygame.K_LEFT:
                vasemmalle = True #liikutaan vasemmalle

            if tapahtuma.key == pygame.K_RIGHT:
                oikealle = True #liikutaan oikealle


        if tapahtuma.type == pygame.KEYUP: #alaspainettu näppäin vapautetaan
            ylos = False  #Lopetetaan 'pelaajan' liikkuminen
            alas = False
            vasemmalle = False
            oikealle = False
            jatketaan = False

        if tapahtuma.type == pygame.QUIT:
                peliJatkuu = False

    if ylos:
        if pelaaja.keskipiste[1] - pelaaja.sade > 0:
            pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] -
2)

    if alas:
        if pelaaja.keskipiste[1] + pelaaja.sade < KORKEUS:
```

```
            pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] +
2)

    if vasemmalle:
        if pelaaja.keskipiste[0] - pelaaja.sade > 0:
            pelaaja.keskipiste = (pelaaja.keskipiste[0] - 2,
pelaaja.keskipiste[1])

    if oikealle:
        if pelaaja.keskipiste[0] + pelaaja.sade < LEVEYS:
            pelaaja.keskipiste = (pelaaja.keskipiste[0] + 2,
pelaaja.keskipiste[1])
```

```
#Peli
#koodi4.py

import pygame
from random import randint #Käytetään satunnaislukuja
from pygame.locals import Rect #Tätä tarvitaan, suorakulmioissa esiintyy 'Rect'
from pygame import mouse

global tauko
#peli jatkuu, kun tämä globaali muuttuja on epätosi

VALKOINEN = (255, 255, 255)
PUNAINEN = (255, 0, 0)
VIHREA = (0, 255, 0)
SININEN = (0, 0, 255)
MUSTA = (0,0,0)
HARMAA = (128,128,128)

LEVEYS = 1000
KORKEUS = 500
REUNASTA = 120 #'maalivahdin' ja 'pelaajan' etäisyys lähimmästä reunasta
alkutilanteessa
OTSIKKO = "Melkein valmis peli"
TEKSTIKENTAN_KORKEUS = 33


class Ympyra:

    def __init__(self, vari, keskipiste, sade):
        self.vari = vari
        self.keskipiste = keskipiste
        self.sade = sade
        self.suunta = (0,0) #Tätä käytetään pallon liikutteluun

    def muuta_suuntaa(self, x,y):
        self.suunta = (x,y)

    def piirra(self):
        global ALUE
        pygame.draw.circle(ALUE,self.vari, self.keskipiste, self.sade, width  = 0)

    def reuna_kosketus(self):
        vasen = (self.keskipiste[0] - self.sade <= 0)
        oikea = (self.keskipiste[0] +  self.sade >= LEVEYS)
        yla = (self.keskipiste[1] - self.sade <= 0)
        ala = (self.keskipiste[1] + self.sade >= KORKEUS)
        return((vasen | oikea), (yla | ala))

    def kimpoaa(self):
        (pystyreuna, vaakareuna) = self.reuna_kosketus()
        if pystyreuna:
            self.muuta_suuntaa(-self.suunta[0], self.suunta[1])
        if vaakareuna:
            self.muuta_suuntaa(self.suunta[0], -self.suunta[1])
```

```
    def siirry(self):
        self.keskipiste = (self.keskipiste[0] + self.suunta[0], self.keskipiste[1]
+ self.suunta[1])


class Suorakulmio:
    def __init__(self, vari, x_sijainti, y_sijainti, leveys, korkeus):
        self.vari = vari
        self.x = x_sijainti
        self.y = y_sijainti
        self.l = leveys
        self.k = korkeus

    def piirra(self):
        global ALUE
        rect = Rect(self.x, self.y, self.l, self.k)
        pygame.draw.rect(ALUE, self.vari, rect, width = 0)

#Määritellään pelin oliot
maalivahti = Ympyra(SININEN, (0,0), 35)
pelaaja = Ympyra(PUNAINEN, (0,0), 35)
pallo = Ympyra(MUSTA, (0,0), 10)
maali_vasen = Suorakulmio(HARMAA, 0, 150, 50, 200)
maali_oikea = Suorakulmio(HARMAA, LEVEYS - 50, 150, 50, 200)


#Pelialueen määrittely
ALUE = pygame.display.set_mode((LEVEYS, KORKEUS + TEKSTIKENTAN_KORKEUS))

VARI = VALKOINEN #Pelikentän taustaväri
ALUE.fill(VARI)

pygame.display.set_caption(OTSIKKO)
pygame.init()

def pelialueen_reunat(viivan_vari, viivan_leveys):
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (LEVEYS, 0), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, KORKEUS), (LEVEYS, KORKEUS),
viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (0, 0), (0, KORKEUS), viivan_leveys)
    pygame.draw.line(ALUE, viivan_vari, (LEVEYS-viivan_leveys, 0),
(LEVEYS-viivan_leveys, KORKEUS), viivan_leveys)

def pelitilanteen_nayttaminen(maalitV, maalitO):
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR = fontti.render(f"Tehdyt maalit: {maalitO}  Omaan maaliin: {maalitV}",
True, MUSTA)
    ALUE.blit(tekstiR, (10, KORKEUS + 5))


def alkuasetelma(maalivahti, pelaaja, pallo, max_x_suunta, max_y_suunta):
    global tauko
    tauko = True
```

```
    maalivahti.keskipiste = (REUNASTA, KORKEUS/2)
    pelaaja.keskipiste = (LEVEYS - REUNASTA, KORKEUS/2)
    pallo.keskipiste = (LEVEYS - REUNASTA, KORKEUS/2)
    mouse.set_pos(maalivahti.keskipiste[0], maalivahti.keskipiste[1])
    #Arvotaan pallolle suunta kohti maalia
    x_suunta = - randint(1,max_x_suunta)
    y_suunta = randint(1,max_y_suunta)
    pallo.muuta_suuntaa(x_suunta, y_suunta)




def pallo_pelialueen_ulkopuolella(pallo):
    x = pallo.keskipiste[0]
    y = pallo.keskipiste[1]
    return (x < 0) | (x > LEVEYS) | (y < 0) | (y > KORKEUS)


def pallo_maalissa(maalivahti, pelaaja, pallo, maali_vasen, maalitV, maali_oikea,
maalitO):
    a = pallo.keskipiste[0]
    b = pallo.keskipiste[1]
    c = maali_vasen.y
    d = maali_vasen.x
    e = maali_oikea.y
    f = maali_oikea.x
    maalissaV = (c < b) & (b < c + maali_vasen.k) & (a < d + maali_vasen.l)
    maalissaO = (e < b) & (b < e + maali_oikea.k) & (a > f)

    if maalissaV:
        maalitV += 1
        ALUE.fill(PUNAINEN)
        alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)

    if maalissaO:
        maalitO += 1
        ALUE.fill(SININEN)
        alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)

    return(maalitV, maalitO)

def maalivahti_pallo(maalivahti, pelaaja, pallo):
    a = maalivahti.keskipiste[0]
    b = maalivahti.keskipiste[1]
    c = pallo.keskipiste[0]
    d = pallo.keskipiste[1]

    if (a-c)**2 + (b-d)**2 <= (pallo.sade + maalivahti.sade)**2:
        #maalivahdilla ja pallolla kosketus
        suunta_x = abs(pallo.suunta[0]) + 2 #itseisarvo, jotta liike kohti oikeaa
reunaa
        if maalivahti.keskipiste[1] < KORKEUS/2:
            suunta_y = 2
        else:
```

```
            suunta_y = -2

        pallo.suunta = (suunta_x, suunta_y) #nopeutta voi säädellä kertomalla tai
jakamalla nämä
        if pallo_pelialueen_ulkopuolella(pallo):
            #Tietyissä tilanteissa maalivahti voi työntää pallon pelialueen
ulkopuolelle.
            alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)



def pelaaja_pallo(pelaaja, pallo):
    #Tässä versiossa ei tarvita parametria maalivahti
    a = pelaaja.keskipiste[0]
    b = pelaaja.keskipiste[1]
    c = pallo.keskipiste[0]
    d = pallo.keskipiste[1]

    if (a-c)**2 + (b-d)**2 <= (pallo.sade + pelaaja.sade)**2:
        pallo.keskipiste = (a - 2*pelaaja.sade, b)
        pallo.muuta_suuntaa(randint(-5,-1), randint(-2,2))
        if pallo_pelialueen_ulkopuolella(pallo):
            alkuasetelma(maalivahti, pelaaja, pallo, 3, 3)



def pelin_alkuteksti():
    global tauko
    ALUE.fill(VALKOINEN)
    fontti = pygame.font.SysFont('arial', 30)
    tekstiR0 = fontti.render("Peli alkaa, kun painat välilyöntiä.", True, MUSTA)
    tekstiR1 = fontti.render("Vasemman puolen maalivahtia liikutetaan hiirellä",
True, MUSTA)
    tekstiR2 = fontti.render("Kun vasen puoli tekee maalin, tulee sininen
välähdys.", True, MUSTA)
    tekstiR3 = fontti.render("Kun oikea puoli tekee maalin, tulee punainen
välähdys.", True, MUSTA)
    tekstiR4 = fontti.render("Peliä voi pelata yksin ja oikean puolen pelaaja
liikkuu itsestään.", True, MUSTA)
    tekstiR5 = fontti.render("Sitä voi kuitenkin liikuttaa myös
nuolinäppäimillä.", True, MUSTA)
    tekstiR6 = fontti.render("Jos vasen tai oikea puoli on ylivoimainen, koodia on
mahdollista", True, MUSTA)
    tekstiR7 = fontti.render("säätää niin, että pelistä tulee tasaisempi.", True,
MUSTA)
    ALUE.blit(tekstiR0, (10, 10))
    ALUE.blit(tekstiR1, (10, 50))
    ALUE.blit(tekstiR2, (10, 90))
    ALUE.blit(tekstiR3, (10, 130))
    ALUE.blit(tekstiR4, (10, 170))
    ALUE.blit(tekstiR5, (10, 210))
    ALUE.blit(tekstiR6, (10, 250))
    ALUE.blit(tekstiR7, (10, 290))
```

```
    pygame.display.update()
    tauko = True
    while tauko:
        for tapahtuma in pygame.event.get():
            if tapahtuma.type == pygame.KEYDOWN:
                if tapahtuma.key == pygame.K_SPACE:
                    tauko = False



def paivita(maalitV, maalitO):
    #Tähän kerätään yhteen kaikki pelialueella tehtävät päivitykset
    global tauko
    ALUE.fill(VARI)
    pelialueen_reunat(PUNAINEN, 3)
    pelitilanteen_nayttaminen(maalitV, maalitO)
    pallo.piirra()
    pallo.kimpoaa()
    pallo.siirry()
    maali_vasen.piirra()
    maali_oikea.piirra()
    maalivahti.piirra()
    #Jos peli on epätasainen, tässä voi säätää
    #pelaajan liikkumisnopeutta kohti palloa.
    if pelaaja.keskipiste[1] < pallo.keskipiste[1]:
      if pelaaja.keskipiste[1] + pelaaja.sade < KORKEUS:
        pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] + 1)
    else:
      if pelaaja.keskipiste[1] - pelaaja.sade > 0:
        pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] - 1)
    pelaaja.piirra()
    maalivahti_pallo(maalivahti, pelaaja, pallo)
    pelaaja_pallo(pelaaja, pallo)
    maalitV, maalitO = pallo_maalissa(maalivahti, pelaaja, pallo, maali_vasen,
maalitV, maali_oikea, maalitO)
    pygame.display.update()
    if tauko:
        tauko = False
        pygame.time.delay(500)
    return(maalitV,maalitO)

kello = pygame.time.Clock()
NOPEUS = 60 #Jos pelaat yksin, nopeuden kasvattaminen vaikeuttaa peliäsi,
#hidastaminen taasen helpottaa.
peliJatkuu = True
ylos = False
alas = False
vasemmalle = False
oikealle = False
torjunnat = 0
maalitV = 0
maalitO = 0
max_x_suunta = 3 #Pallon nopeutta voi säädellä
max_y_suunta = 3 #näillä. Isompi luku johtaa todennäköisesti nopeampaan
```

```
liikkumiseen

pelin_alkuteksti()
alkuasetelma(maalivahti, pelaaja, pallo, max_x_suunta, max_y_suunta)
while peliJatkuu:
    kello.tick(NOPEUS)
    maalitV, maalitO = paivita(maalitV, maalitO)

    for tapahtuma in pygame.event.get():
        if tapahtuma.type == pygame.MOUSEMOTION:
                uusi_keskipiste = (tapahtuma.pos[0], tapahtuma.pos[1])

                maalivahti.keskipiste = uusi_keskipiste

        if tapahtuma.type == pygame.KEYDOWN: #jotain näppäintä on painettu

            if tapahtuma.key ==  pygame.K_UP:  #Jos näppäin oli 'nuoli ylös'

                ylos = True  #liikutaan ylos

            if tapahtuma.key ==  pygame.K_DOWN: #Jos näppäin oli 'nuoli alas'

                alas = True #liikutaan alas

            if tapahtuma.key == pygame.K_LEFT:
                vasemmalle = True #liikutaan vasemmalle

            if tapahtuma.key == pygame.K_RIGHT:
                oikealle = True #liikutaan oikealle


        if tapahtuma.type == pygame.KEYUP: #alaspainettu näppäin vapautetaan
            ylos = False  #Lopetetaan 'pelaajan' liikkuminen
            alas = False
            vasemmalle = False
            oikealle = False
            jatketaan = False

        if tapahtuma.type == pygame.QUIT:
                peliJatkuu = False

    if ylos:
        if pelaaja.keskipiste[1] - pelaaja.sade > 0:
            pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] -
2)

    if alas:
        if pelaaja.keskipiste[1] + pelaaja.sade < KORKEUS:
            pelaaja.keskipiste = (pelaaja.keskipiste[0], pelaaja.keskipiste[1] +
2)

    if vasemmalle:
        if pelaaja.keskipiste[0] - pelaaja.sade > 0:
            pelaaja.keskipiste = (pelaaja.keskipiste[0] - 2,
```

```
pelaaja.keskipiste[1])

    if oikealle:
        if pelaaja.keskipiste[0] + pelaaja.sade < LEVEYS:
            pelaaja.keskipiste = (pelaaja.keskipiste[0] + 2,
pelaaja.keskipiste[1])
```

Karkausvuosi-ohjelmointia, 9. lk

1.4.2022

Tämä suunnitelma on tehty 9. luokan oppilaille, jotka ovat jo tehneet Pythonilla jonkin verran ohjelmointia. Tehtävä toteutetaan matematiikan tunneilla, aikaa varataan kaksi 45 minuutin oppituntia.

Tehtävän tavoitteena on tutustua karkausvuoden määritelmään. Lisäksi tehtävässä ohjelmoidaan ohjelma, joka kysyy vuosilukua ja kertoo onko kyseinen vuosi karkausvuosi.

Suunnitelma:

Ensimmäinen oppitunti:

1. Oppilaat jakautuvat 2-3 hengen ryhmiin. Tämän jälkeen oppilaat etsivät Internetistä tiedon miten karkausvuosi määritetään. (10-15 min)

2. Käydään yhdessä läpi määritelmä ja ohje mitä ohjelmoinnin tuloksena pitäisi syntyä. (10-15 min)

3. Oppilaat suunnittelevat ohjelman algoritmia paperille. (10-15 min)

Toinen oppitunti:

1. Oppilaat lähtevät yhdessä kirjoittamaan koodia tekemänsä algoritmin pohjalta ja testaavat toimiiko ohjelma halutulla tavalla. (noin 30 minuuttia)

2. Arviointi. Opettaja käy keskustelemalla läpi jokaisen ryhmän kanssa miten tehtävän tekeminen sujui. (noin 15 minuuttia)

Mallikoodi:

```
vuosi = int(input("Anna vuosiluku"))
jakojaannos400 = vuosi % 400

jakojaannos100 = vuosi % 100

jakojaannos4 = vuosi % 4

if  jakojaannos4 == 0 and jakojaannos100 == 0 and jakojaannos400 == 0:
  print("Kyseinen vuosi on karkausvuosi")
elif jakojaannos4 == 0 and jakojaannos100 > 0 and jakojaannos400 > 0:
  print("Kyseinen vuosi on karkausvuosi")
else:
  print("Kyseinen vuosi ei ole karkausvuosi")
```

Python-ohjelmointia Tie koodariksi -sivuston avulla, 9. lk

Johanna Vaurasalo, CC BY-SA 4.0

15.10.2022

Algoritmisen ajattelun kehittäminen: Ohjelmoinnin opetuksen suunnittelu

TIE KOODARIKSI -tuntisuunnitelma

Yläkoulussa on tavoitteena antaa oppilaille mahdollisuus tekstipohjaisen ohjelmoinnin opiskeluun. Oppilailla tuntuu olevan aika vähän kokemusta ohjelmoinnista, joten ohjelmoinnin perusasioista on lähdettävä liikenteeseen. Tähän ohjelmoinnin perusteiden harjoitteluun sopii erinomaisesti Tie koodariksi -tehtävät matematiikan tunneilla. Tehtävien avulla kerrataan ja harjoitellaan matematiikan taitoja sekä opitaan ohjelmointia Python-kielellä. Python-kieli on yksi suosituimmista ohjelmointikielistä, joten sekin tukee näiden tehtävien valintaa. Ohjelmointiharjoituksissa kehittyy lisäksi looginen ajattelu, ongelmanratkaisutaidot sekä tietokoneen käyttömahdollisuuksien uudenlainen ymmärtäminen.

Opettaja saa järjestelmään opettajan oikeudet, joiden avulla hän pystyy seuraamaan oppilaiden edistymistä tehtävissä ja näkemään tehtävien malliratkaisut. Opettaja saa oikeudet luomalla ensin tunnuksen sivustolle ja lähettämällä sitten viestin osoitteeseen ahslaaks@cs.helsinki.fi ja kertomalla viestissä tunnuksensa sekä missä koulussa opettaa.

Jokaisella oppilaalla tulee olla tunnilla tietokone. Oppilas liittyy ryhmään opettajalta saadulta ryhmäavainkoodilla. Tehtävissä lähdetään liikkeelle aivan alusta, eikä oppilailla tarvitse olla aiempaa kokemusta ohjelmoinnista. Tehtävät muodostuvat 16 luvusta, joihin kuuluu automaattisesti tarkastettavia tehtäviä. Ne on toteutettu niin, että oppilaat voivat ohjelmoida suoraan nettiselaimessa eikä heidän tarvitse asentaa koneelle ohjelmaa. Tehtävät soveltuvat hyvin myös itseopiskeluun, joten oppilaat voivat tarvittaessa tehdä niitä vaikka kotona, jos ovat olleet poissa tunnilta.

Tunnin kulku

Tämä ohjelmointitunti on tarkoitus toteuttaa 9. luokan matematiikan kaksoistunnilla. Tehtävät liittyvät kiinteästi matematiikkaan, joten johdantona ja motivointikeinona tunnin alussa kerron, että tunnilla on tarkoitus kerrata matematiikkaa hieman aiempaa poikkeavalla tavalla. Kynää ja paperia voi käyttää apuna, mutta tehtävät tehdään kuitenkin tietokoneelle ohjelmoimalla. Lisäksi käydään yhdessä läpi Tie koodariksi -sivuston ensimmäinen luku eli ohjelmoinnin historia. Näytän oppilaille ryhmän avainkoodin, jolla he liittyvät Tie koodariksi -kurssin sivulle.

Koulussamme on käytössä pisteiden kerääminen tehtyjen tehtävien mukaan. Tästä harjoituksesta saa myös pisteitä. Jokaisesta perustehtävästä saa yhden pisteen kuten kirjan perustehtävistäkin. Sen sijaan vaativammista tehtävistä saa kaksi pistettä ja kaikkein vaativimmista ohjelmointitehtävistä jopa kolme tai neljä pistettä. Tehtäviä pisteyttämällä oppilaat saadaan tuntemaan, että nämä ohjelmointitehtävät kuuluvat osaksi matematiikan opiskelua, joka tuottaa heille samalla tavalla pisteitä kuin normaalit kirjan tehtävät. Tosin kolmen ja neljän pisteen tehtäviä kirjassa ei ole, joten taitavat oppilaat saavat niistä extrapisteitä, mutta jospa se motivoisi heitä yrittämään entistä kovemmin.

Rohkaisen oppilaita tekemään yhteistyötä ja auttamaan toisiaan kuten normaaleilla matikantunneillakin. Tehtäviä voi tehdä yksin, parityönä tai ryhmissä, mutta jokainen tekee tehtävät omalla koneellaan, jolloin jokaiselle kertyy myös pisteet ja näyttöä siitä, että ovat harjoitelleet ohjelmointia ja kerranneet matematiikkaa. Oppilaat voivat edetä tehtävissä omaan tahtiin. Tavoitteena on tehdä lukujen 2–4 tehtävät. Oppilaat saavat tehtävistä välittömän palautteen ohjelman avulla ja uskon, että tehtävissä onnistuminen koukuttaa heitä yrittämään parhaansa.

Taitavat koodarit voivat jatkaa niin pitkälle kuin ehtivät ja saada sitä kautta kerättyä itselleen lisäpisteitä. Se voi olla heille mieluisampi tapa kerätä matematiikan pisteitä kuin tutut vihkotehtävät. Jos joku lukujen 2–4 tehtävistä ei onnistu kaikilta oppilailta, niin ne oppilaat, jotka ovat tehtävän osanneet voisivat käydä neuvomassa tehtävien tekemisessä. Kaksoistunnin lopussa kaikilla olisi hyvä olla kyseiset tehtävät tehtynä. Jos näyttää siltä, etteivät oppilaat saa apua edistyneemmiltä koodareilta, niin opettaja voi käydä heidän kanssaan puuttuvat tehtävät läpi.

Tunnin lopussa voitaisiin pitää yhteinen palautekeskustelu siitä, miltä tehtävät tuntuivat ja onko oppilailla kokemusta jostain muusta ohjelmointikielestä. Jos on, niin mistä kielestä ja miten nämä kielet eroavat toisistaan.

Lähteet:

https://tie.koodariksi.fi/alkeet/

https://fi.wikipedia.org/wiki/Python_(ohjelmointikieli)

https://tie.koodariksi.fi/alkeet/1

Micro:bit, yläkoulu

Sanna Toivanen, CC BY-SA 4.0

5.5.2019

TUNTISUUNNITELMA **LUMATIKKA – algoritmisen ajattelun kehittäminen**

Tunnit on suunniteltu matematiikkaan yläkoulun oppilaille, joilla ei välttämättä ole paljoa kokemusta ohjelmoinnista. Tunneilla käytetään olemassa olevia Innokas-materiaaleja (https://www.innokas.fi/materiaalit/?_sft_material_type=microbit). Tarkemmat ohjeet löytyvät tuntisuunnitelman lopusta. Harjoitukset vievät 2-3 oppituntia ryhmästä ja oppitunnin pituudesta riippuen. Tunnit on liitetty osin todennäköisyyksien opiskeluun.

1. tehtävän tavoitteena on tutustua micro:bitin käyttöön (ohjelmointi, koodin siirto ja suoritus) sekä toistorakenteeseen.

2. tehtävän tavoitteena on koodata micro:bitin avulla noppa, jolla määritetään eri silmälukujen tilastollisia todennäköisyyksiä ja verrataan niitä klassiseen todennäköisyyteen.
Nopan koodaamista varten kerrataan muuttujan käsite. Noppaa ”heittämällä” tarkastelemme tilastollista ja klassista todennäköisyyttä. Eli kokoamme jokaisen oppilaan tilastolliset tulokset taulukkoon ja vertaamme tuloksia klassiseen todennäköisyyteen. Tarkastelu tehdään keskustelemalla yhdessä.

3. tehtävänä tehdään matikkapeli, jossa hyödynnetään micro:bitin eri ominaisuuksia ja käytetään useita muuttujia ja matematiikkatyökaluja. Tavoitteena on ymmärtää muuttujan merkitys, peli harjoittaa myös päässälaskua. Jokaisella oppilaalla on oma micro:bit, mutta he voivat tehdä myös yhteistyötä keskustelemalla sekä testaavat pareittain pelinsä toimivuutta. Pelin ideana on arvata kaverin ajattelema luku, kun oppilas ja kaveri syöttävät micro:bittiin ajattelemansa luvut micro:bitin tulostaessa lukujen summan. Ohjeen mukaan koodatessa ohjelma ei nollaa summaa, vaan summa kasvaa joka pelikerralla. Tämän vuoksi ohjeissani on oppilaille lisäkysymyksiä ohjelmakoodin muuttamista varten. Oppilailla on mahdollisuus myös ideoida oma peli.

Opettajan arviointi perustuu työskentelyn tarkasteluun ja keskusteluihin. Oppilas saa osaamisestaan palautetta ohjelman toimivuudesta sekä (luokkakavereilta ja) opettajalta keskusteluissa. Tehtävien ohjeissa on aluksi esitetty ohjelman ajatus, joten ohjelmakoodia voi miettiä itse (kokeilemalla oppiminen). Tehtävissä on myös yksityiskohtaiset ohjeet, joten jokaisen pitäisi saada niillä ohjelma tehtyä.

Välineet:
Micro:bit https://microbit.org/fi/
BBC micro:bit on ARM-pohjainen kehitysalusta, jonka Britannian yleisradioyhtiö BBC on kehittänyt koulujen tietotekniikan opetukseen. Kooltaan se on 43 mm x52 mm. Prosessorin lisäksi piirilevyllä on 25:stä ledistä muodostuva näyttö, led-valoja, joiden toimintaa voidaan ohjata. Lisäksi laitteessa on mm. sekä kolme painiketta sekä kiihtyvyys- ja magneettikenttäanturit, joiden avulla voidaan tunnistaa painalluksia, liikettä ja liikesuuntia. Laitteeseen on mahdollista kiinnittää myös ulkoisia antureita.

- Antaa konkreettisen kokemuksen ohjelmoinnista
- Voi ohjelmoida sekä graafisella että tekstipohjaisilla ohjelmointikielillä
- Runsaasti sovellusmahdollisuuksia, mahdollistaa maker-kulttuuria
- Tarvitaan omat micro:bitit sekä ohjelmointia varten tietokoneen tai mobiililaitteen

Yleiset ohjeet:

- Yhdistä BBC micro:bit tietokoneeseen USB-kaapelilla. (Voi käyttää ohjelmointiin tietokoneen sijaan myös mobiililaitetta, jolloin täytyy ladata micro:bit-sovellus.)
- Ohjelmointi: Kirjoita oma ohjelmakoodi osoitteessa https://microbit.org/code/. Voit ohjelmoida joko graafisesti koodipalikoiden avulla https://makecode.microbit.org/# tai käyttää tekstipohjaisia editoreja https://python.microbit.org/v/1.1.
- Siirto: Kun koodi on valmis, lataa se tietokoneelle ja kopioi micro:bitille. Mikäli käytät sovellusta, lataa koodi sovellukseen ja lähetä se Bluetooth-yhteyden avulla micro:bitille.
- Ohjeman suoritus: Kun koodi on siirtynyt kokonaan, micro:bit lähtee automaattisesti suorittamaan koodia.

https://fi.wikipedia.org/wiki/Micro_Bit
https://ilonait.fi/wp-content/uploads/2017/08/Microbit_pedakortti_web.pdf

Materiaalit:
Käytetään olemassa olevia Innokas-materiaaleja.
https://www.innokas.fi/materiaalit/?_sft_material_type=microbit
Ohjeet jaetaan oppilaille sähköisesti esim. käytettävissä olevassa oppimisympäristössä, (opetusryhmän O365-Teams). Ohjeet ovat tarkoitettu ohjelmointiin visuaalisella editorilla (MakeCode Editor), mutta edistyneet oppilaat voivat tehdä ohjelmat myös tekstipohjaisella Python-editorilla.

Ohjeet:
Mene tietokoneen selaimella linkin osoitteeseen ja tee tehtävä ohjeen mukaan.

1. harjoitus: Toistolause https://www.innokas.fi/wp-content/uploads/2018/12/02_Microbit_tehtavakortti_toistolause-PA%CC%88IVITETTY.pdf
- sisältää aloitusohjeet, tutustutaan mikrobitin käyttöön ja ohjelman siirtämiseen koneelta mikrobittiin ja ohjelman suorittamiseen

2. harjoitus: muuttuja(noppa) https://www.innokas.fi/wp-content/uploads/2018/12/03_Microbit-Muuttuja-Noppa-PA%CC%88IVITETTY.pdf
- luodaan noppa, jonka avulla tarkastelemme todennäköisyyksiä.
- Kun saat nopan toimimaan, ”heitä” noppaa 20 kertaa ja kirjaa tulokset taulukkoon (*linkki*). Tarkastele tuloksia. Vastaavatko tulokset klassista todennäköisyyttä?
- jos aikaa on, niin todennäköisyyksien tarkastelun jälkeen jatka harjoituksen lisätehtäviä. Voit vapaasti muuttaa ohjelmaa ja koodata oman noppaohjelman.

3. harjoitus: salaluku-matikkapeli https://www.innokas.fi/wp-content/uploads/2018/02/15_Microbit-matikkapeli-FINAL.pdf
- jokainen ohjelmoi oman summa-matikkapelin, testaa kaverin kanssa kummankin pelin toimivuus

    - miten ohjelma toimii, jos pelataan useampi peli? Miten haluat sen toimivan? Pitäisikö/voisiko koodia muuttaa tässä tilanteessa?
- (lisätehtävänä) tee vastaava kertolaskupeli
- (lisätehtävänä) voit keksiä oman pelin tai tee halutessasi kivi-paperi-sakset peli omin päin tai seuraavalla ohjeella:
  KiviPaperiSakset-peli https://www.innokas.fi/wp-content/uploads/2018/12/Microbit-kivi-paperi-sakset-PA%CC%88IVITETTY.pdf

Ehtolauseiden harjoittelua Pythonilla, yläkoulu

Heini Soppi, CC BY-SA 4.0

6.1.2020

Algoritmisen ajattelun kehittäminen

Tuntisuunnitelma

Heini Soppi

Ehtolauseiden harjoittelua Pythonilla

Aihepiirin valinta ja tavoite

Tarkoitus on harjoitella ehtolauseiden käyttöä Pythonilla ohjelmoimalla pieni tietovisa. Oppiaine on matematiikka ja harjoitellaan ihan puhtaasti ohjelmointia, joskin toki samalla tulee harjoitettua algoritmista ajattelua ja loogista päättelyä. Oppituntien tavoitteena on oppia ymmärtämään ehtolauseiden idea (toisensa poissulkevat vaihtoehdot, joista seuraa erilaisia tapahtumia) ja myös oppia kirjoittamaan hyvien ohjelmointikäytäntöjen mukaista koodia, jossa esiintyy ehtolauseita. Harjoitukseen on mahdollista liittää jonkin toisen oppiaineen sisältöjä valitsemalla tietovisan aiheeksi vaikkapa ruotsin sanasto tai Suomen historian tärkeät vuosiluvut. Tietovisan aihe voidaan jättää myös oppilaiden vapaasti suunniteltavaksi.

Oppimistavoitteet ja niiden arviointi

Tavoitteena on, että oppituntien jälkeen oppilas:

- ymmärtää, mitä ovat ehtolauseet

- osaa kirjoittaa Pythonilla ehtolauseita sisältävää koodia

- on toteuttanut Pythonilla pienen tietovisan, jonka koodi on myös vertaisarvioivalle oppilaalle/työparille ymmärrettävä, ja joka on käyttäjälle ystävällinen

Oppimistavoitteiden saavuttamista arvioidaan kolmella tapaa. Oppilaat palauttavat valmiin työn opettajalle, joka arvioi työn arviointimatriisin avulla (liite 1). Oppilaat arvioivat myös itse omaa työskentelyään ja osaamistaan itsearviointilomakkeella (liite 2). Lisäksi toinen oppilas/työpari arvioi koodin sekä visan toimivuutta ja ymmärrettävyyttä vertaisarviointilomakkeella (liite 3).

Työskentelyvälineet ja opetusmenetelmät

Opetus tapahtuu tietokoneilla. Pythonia käytetään Trinket-ympäristössä (trinket.io), jonne oppilaat ovat jo aiemmin luoneet itselleen käyttäjätilit, tai muussa sopivassa ympäristössä. Harjoituksen voi tehdä ryhmästä, tietokoneiden määrästä ja opettajan toiveista riippuen pareittain tai yksin.

Oppilaita rohkaistaan pohtimaan ja ratkaisemaan asioita yhdessä myös siinä tapauksessa, että jokainen tekee oman tuotoksen.

Ennakkotiedot aiheesta

Jo aiemmin on luotu käyttäjätilit Trinketiin ja harjoiteltu Pythonin alkeita. Ennen tätä harjoitusta oppilaiden oletetaan osaavan:

- käyttää print ja input -komentoja

- ymmärtää muuttujan käsite ja olla harjoitellut muuttujien ohjelmointia
- muuttaa muuttujia muodosta toiseen (int ja str)
- kommentoida ohjelmakoodia

Tuntien kulku

Tarvittaessa voidaan aluksi lyhyesti kerrata aiemmin opittuja asioita. Koko ryhmän kanssa yhdessä pohditaan ensin ehtolauseiden olemassaoloa ja rakennetta. Tässä vaiheessa ei tarvitse mennä vielä varsinaisesti ohjelmointiin, vaan voidaan pohtia asiaa yleisemmin ja käyttää vaikkapa erilaisia kaavioita havainnollistamiseksi. Käydään myös läpi if-elif-else -rakenne ja komennot, jotta oppilaat osaavat kirjoittaa ne oikein (sisennykset ym.).

Oppilaille selitetään harjoitustyön idea ja annetaan työohje sekä kerrotaan arvioinnista. Tarkoitus on, että he saavat pohtia ja edetä asiassa itsenäisesti omaan tahtiin, mutta toki opettaja ohjaa työskentelyä tarpeen mukaan. Työohje on tarkoituksella melko avoin, jotta oppilaiden omalle oivaltamiselle jää tilaa.

Oppilaiden tulisi ratkaista joko itse tai johdateltuna seuraavat asiat:

- miten pelaajaa ohjeistetaan vastaamaan kysymyksiin?
- missä muodossa pelaaja vastaa kysymyksiin?
- miten pelaajan vastaus voidaan muuttaa luvuksi?
- miten ehtolauserakenne toimii?
- millainen ehto oikealle vastaukselle on? Mitä ehdon täyttymisestä seuraa? Mitä seuraa väärästä vastauksesta?
- miten luodaan muuttuja pistelaskua varten? Miten tämä muuttuja saadaan kasvamaan oikeiden vastausten myötä?
- miten pelaajalle voidaan ilmoittaa oikeiden vastausten lukumäärä?

Valmis työ voidaan palauttaa kuvankaappauksina tai tekstitiedostona opettajalle tai esim. Google Classroom -ryhmään sen mukaan, mitä tapaa koulussa käytetään. Opettaja arvioi työn arviointimatriisin avulla. Oppilas täyttää itsearviointilomakkeen. Vertaisarviointi voidaan tehdä niin, että vertaisarvioivat oppilaat kokeilevat tietokoneella toistensa tietovisaa ja käyvät koodin läpi sekä täyttävät arviointilomakkeen.

Työohje

Tehtävänäsi on ohjelmoida tietovisa, jossa on vähintään kolme kysymystä vastausvaihtoehtoineen. Kirjoita ohjelma sellaiseksi, että se esittää kysymyksen ja kysyy käyttäjältä vastausta sekä kertoo, onko vastaus oikein vai väärin. Käytä koodissa ehtolauseita. Visan lopussa ohjelman tulisi kertoa, montako pistettä käyttäjä sai.

Aloita pohtimalla, mitä kaikkia tietoja tarvitset voidaksesi ohjelmoida tietovisan. Millaisista ”palasista” koodisi tulee koostumaan?

Mieti ohjelmoidessasi seuraavia asioita (nämä huomioidaan myös arvioinnissa):

- Pyri kirjoittamaan koodi niin, että vertaisarvioijakin ymmärtää, mitä olet tehnyt. Kommentoi koodia ja kirjoita se mahdollisimman selkeästi.
- Tee pelistä sellainen, että sitä on miellyttävä pelata. Mieti, mitä ohjeita peli antaa pelaajalle ja millaista palautetta pelaaja saa pelistä.

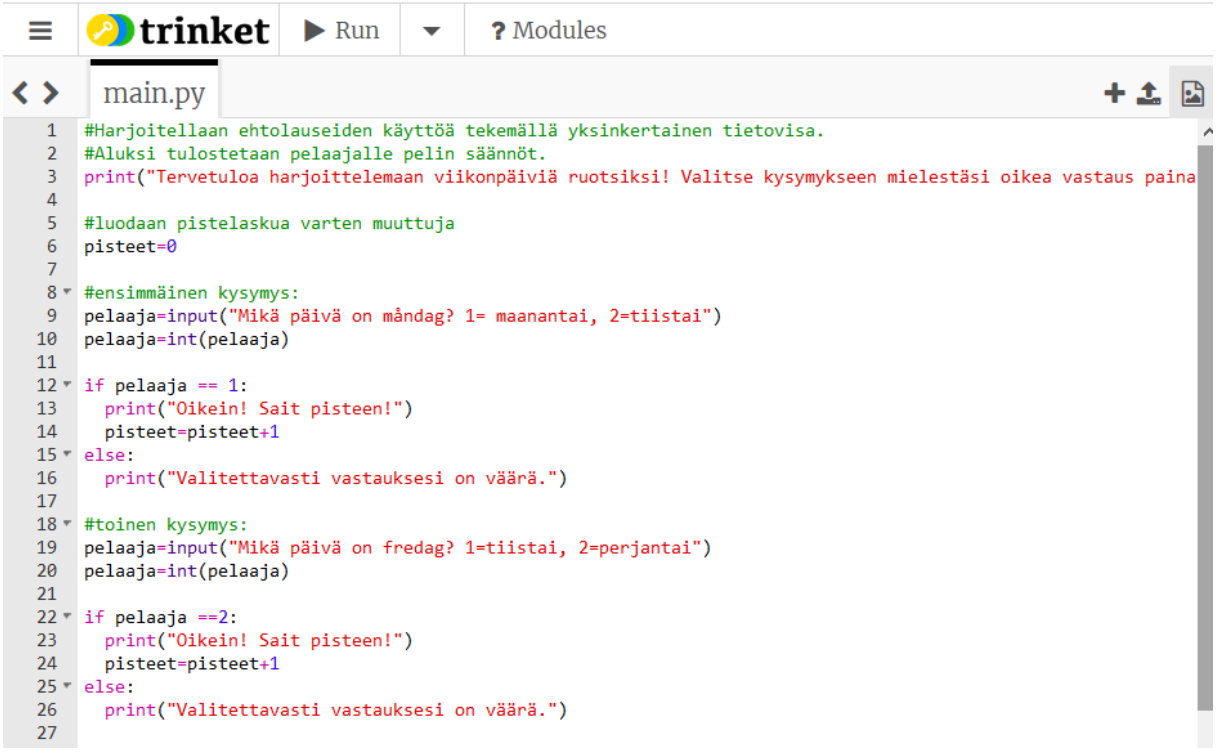
Jos jäät jumiin tai alkuun on vaikea päästä, voit tehdä aluksi vaikka paperille kaaviona hahmotelman siitä, miten peli vaihtoehtoineen etenee.

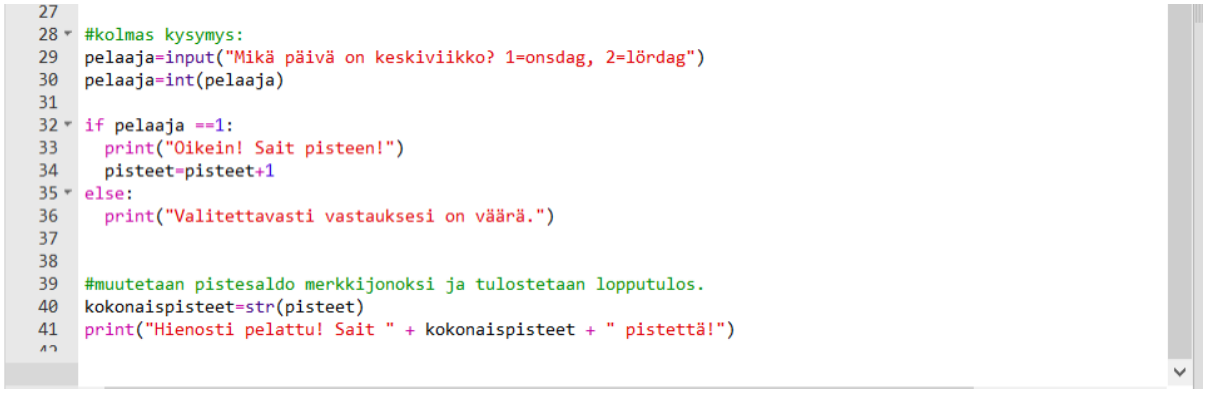
Eriyttäminen

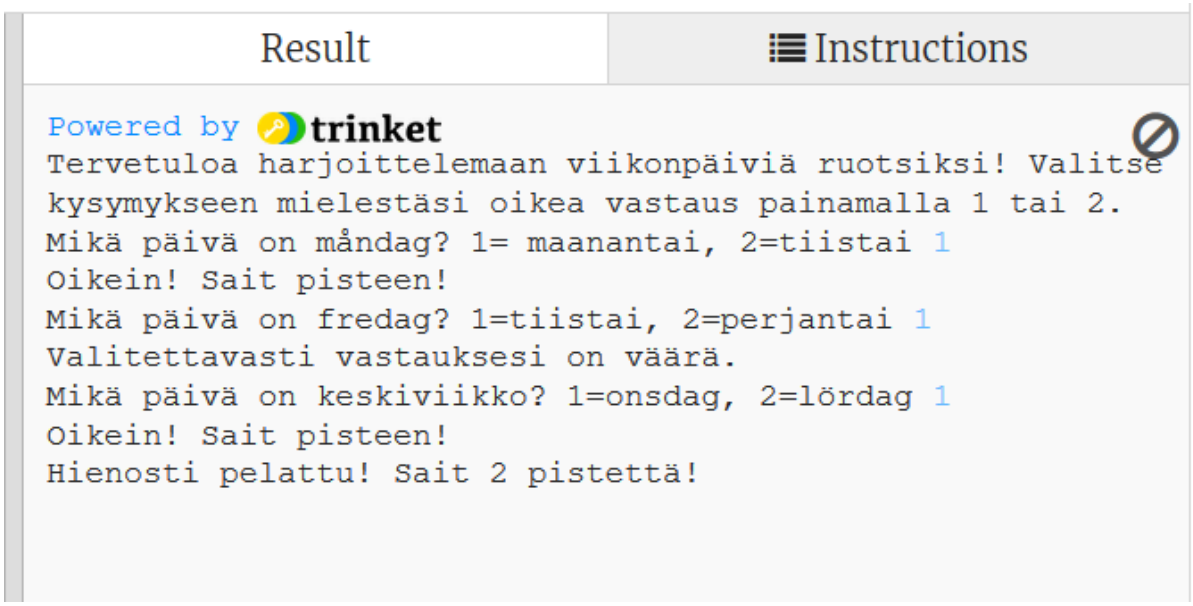
Tehtävää voi eriyttää alaspäin esimerkiksi jättämällä pistelaskun kokonaan pois. Tarvittaessa tehtävää on helppo eriyttää myös ylöspäin. Oppilaat voivat miettiä, olisiko visan kysymyksiin enemmän kuin kaksi vaihtoehtoa niin, että peli antaisi jokaisesta vaihtoehdosta erilaisen palautteen pelaajalle (if-elif-else -rakenne). Kokeneemmat ohjelmoijat voivat kehittää peliä myös niin, että pelaajan antaman vastauksen jälkeen ohjelma tarkistaa, annettiinko vastaus oikeassa muodossa ja ilmoittaa, jos vastauksen tyyppi on väärä.

Esimerkki valmiista koodista

Kuvankaappauksissa esimerkki tehtävänannon mukaisesta koodista ja siitä, miltä visa näyttää pelattuna. Tässä aiheeksi on valittu viikonpäivät ruotsiksi.







Liite 1: Arviointimatriisi opettajan suorittamaan arviointiin

	ei lainkaan	jonkin verran / pieniä virheitä	hyvin
Koodi toimii ilman virheilmoituksia			
Koodissa on käytetty ehtolauseita oikein			
Pistelasku muuttujan avulla toimii			
Koodia on kommentoitu			
Koodi on ymmärrettävä, ei turhia osia			
Pelaaja saa ohjeistuksen peliin			

Liite 2: Itsearviointilomake

	Ei lainkaan	Jonkin verran	Hyvin
Käytin ohjelmassa ehtolauseita			
Sain pistelaskun toimimaan			
Sain tietovisan toimimaan			
Koodini on ymmärrettävää			
Muistin kommentoida koodia			
Pelaaja saa ohjelmalta tarvittavat ohjeet			
Ymmärrän itse, mitä olen ohjelmoinut			

Mitä opit tästä työstä?

Missä kohdissa oli vaikeuksia?

Millainen oli työnjakonne, jos teitte työn ryhmässä?

Liite 3: Vertaisarviointi

	Ei lainkaan.	Jollain lailla.	Hyvin.
Pystyin pelaamaan tietovisan (peli toimi).			
Peli antoi ohjeet ja kertoi, onko vastaukseni oikea.			
Sain lopussa tietää pisteeni.			
Luin koodin ja ymmärsin, mitä siinä oli tehty.			

Muuta palautetta pelistä tai koodista:

Ohjelmoinnin opiskelu aoe.fi-sivuston materiaaleja hyödyntäen, yläkoulu

Tero Toivanen, CC BY-SA 4.0

6.1.2020

Ohjelmoinnin opiskelu aoe.fi-sivuston materiaaleja hyödyntäen

Aihepiirin valinta ja rajaus

Vaikka ohjelmointi mainitaan hyvän osaamisen kriteereissä sekä matematiikassa että käsitöissä, se mielestäni kuuluu kaikkiin oppiaineisiin. Itse asiassa on mielestäni sääli, jos se jäisi vain näiden oppiaineiden sisällöksi. Koska käytän valmiita avoimia oppimateriaaleja hyväksi, voin helposti eriyttää tehtävän eri oppilaiden lähikehityksen vyöhykettä hyödyntäen ja mahdollistaen yksilöllisen etenemisen. **Aiheena** on siis ohjelmoinnin peruskäsitteet niin, että tavoitteena olisi oppia käyttämään ja hyödyntämään sekä lauseita että silmukoita. Nopeimmat voivat edetä edelleen ehtolauseisiin, muuttujiin ja jopa listoihin. Käytössä on sekä Scratch- että Python -ohjelmointikielet sekä niiden avoimet oppimateriaalit aoe.fi-sivustossa. **Tavoitteena** on oppia tekemään toimiva ohjelma joko graafisessa Scratch- tai sitten Python -ympäristössä. Saman luokan sisällä löytyy oppilaita eri luokka- ja taitotasoilta, joten eriyttävä materiaali on kullan arvoista.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppimistavoitteena on oppia käyttämään lauseita ja silmukoita ohjelmaa rakennettaessa tietoisesti. **Kohdeoppiaineena** ovat kaikki oppiaineet eli opittua voi soveltaa missä oppiaineessa hyvänsä tai sitten esimerkiksi monialaisen oppimiskokonaisuuden toteutuksessa.

Arviointi tapahtuu formatiivisena arviointina niin, että opettaja tukee oppilaiden ongelmanratkaisua oikea-aikaisella auttamisella scaffoldingin periaatteiden mukaisesti. Oppilaat myös dokumentoivat omia projektejaan niin, että projektin vaiheet tulee kirjattua (tässä opettajan tuki ja kannustus voivat olla todella tarpeen). **Oppilaat arvioivat** omaa työskentelyään ja sitä kuinka heidän ohjelmointitaitonsa kehittyvät prosessin aikana. Arvioinnin kohteena ovat siis ohjelmointiosaaminen sekä ongelmanratkaisutaidot (algoritminen osaaminen).

Työskentelyvälineet ja opetusmenetelmät

Oppikokonaisuudessa hyödynnetään aoe.fi-sivustolta löytyviä valmiita materiaaleja. Koska luokassa on hyvin eritasoisia ohjelmoinnin opiskelijoita käytetään hyväksi sekä Scratch- että Python -materiaaleja. Materiaaleissa on lähes identtiset tehtävät molemmille materiaaleille, joten ne soveltuvat erinomaisesti eriyttävään lähestymistapaan.

Opettaja toimii **johdantomateriaalin** esittelijänä. Johdantomateriaalina toimii aina edellä mainittujen sivustojen teoriaosuudet kuten Lause Scratchissä ja Lause Pythonissa

-materiaalit. Opettaja toimii myös koko ajan fasilitaattorina ja oppimisen oppaana prosessissa. Ohjelmointiympäristöinä toimivat Schratchissä scratch.mit.edu, joka on linkitetty suoraan materiaaleihin sekä trinket.io Pythonissa, joka on upotettuna sivustossa.

Ohjelmointitehtävät ovat haasteita ja niissä opitaan samalla animaation ja pelin tekemistä. Nämä asiat ovat monille oppilaista itsessään **motivoivia**. Turtle-kirjaston hyödyntäminen on mahdollistanut ohjelmoinnin opiskelun aloittamisen suoraan animaatioita hyödyntäen myös Python-ympäristössä, jota perinteisesti on aloitettu opiskelemaan tekstiä ja numeroita käsittelemällä.

Oppilaita kannustetaan **yhteistyöhön** ja parityöskentelyä suositaan. Pidemmällä olevat oppilaat voivat myös täysin vapaasti tukea alussa olevia oppilaita ja näin syventää omaa osaamistaan samalla muita auttaen.

Oppilaiden aikaisempi **ohjelmointiosaaminen** ratkaisee sen, minkä materiaalin kanssa aloitetaan.

Tässä ollaan mainittu kaksi oppituntia, joiden sisältönä olisi siis vähintään Lause ja Silmukka. Tarkoituksena on kuitenkin jatkaa koko kurssi läpi niin, että päästään jatkamaan vaativimpiin projekteihin ja oppisisältöihin. Tämän jälkeen oppilaat voivat **soveltaa osaamistaan itseään motivoiviin ja muihin tarpeellisiin asioihin**. Algoritminen ajattelu tukee myös kaikkea oppimista ja antaa työkaluja selvitä monista eri ongelmanratkaisutilanteista. Itseohjautuvuuden kehittymisen myötä oppilaiden oppimistaidot lisääntyvät ja valmiudet kohdata tulevia haasteita paranevat.

Alkuluvut ja Eratostheneen seula, yläkoulu

11.6.2020

Alkuluvut ja Eratostheneen seula

Määritelmä: Alkuluku on lukua 1 suurempi luonnollinen luku, joka ei ole jaollinen muilla positiivisilla kokonaisluvuilla kuin yhdellä ja itsellään.

1) Mainitse kolme lukua, jotka ovat alkulukuja ja kolme lukua, jotka eivät ole alkulukuja.

   Ovat alkulukuja:

   Eivät ole alkulukuja:

Eratostheneen seula käsin

2) Ympyröi seuraavaan taulukkoon kaikki lukua 100 pienemmät alkuluvut.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Vinkki: Luku 2 on pienin alkuluku. Ympyröi se ja poista kaikki sen monikerrat. Seuraava alkuluku on 3. Ympyröi se ja poista kaikki sen monikerrat. Jne.

Lukua sata pienempiä alkulukuja ovat:

Katso tarvittaessa tarkistus: Eratostheneen seula

Eratosheneen seula tietokoneella

Tutustuminen for-silmukkaan

3) Testaa, mitä seuraavat ohjelmat tekevät:

```
for i in range(10):
    print(i)
```

```
for i in range(1,10):
    print(i)
```

```
for i in range(1,10,2):
    print(i)
```

4) Miten tulostat luvut 1-100?

5) Miten tulostat luvut 50-100?

6) Miten tulostaisit luvut 2, 4, 6, … 100?

7) Mitä seuraava ohjelma tekee?

```
for i in range(1,11):
    print("Luvun",i,"nelio on",i*i)
```

8) Miten tulostaisit lukujen 1-5 kuutiot?

9) Mitä seuraava ohjelma tekee?

```
for a in range(1,6):
    for b in range(1,11):
        print(a*b)
```

Tutustuminen listoihin

10) Mitä seuraavat ohjelmat tekevät?

```
lista=[2,3,4,5,6]
print(lista)
```

```
lista=[]
lista.append(1)
lista.append(2)
lista.append(3)
lista.append(4)
lista.append(5)
print(lista)
```

```
i=0
lista=[i]*5
print(lista)
```

```
lista=[]
for i in range(0,10):
    lista.append(i)
print(lista)
```

```
lista=[]
for i in range(0,10):
    lista.append(i)
lista.remove(5)
print(lista)
```

```
n=20
i=2
lista=[]
for i in range(2,n+1):
    lista.append(i)
print(lista)
for i in range(2*i,n+1,i):
    if i in lista:
        lista.remove(i)
print(lista)
```

11) Eratostheneen seulan ohjelmointi

    Tee ohjelma, joka tulostaa lukua 100 pienemmät alkuluvut.

    Vihje: Muodosta lista luvuista 2-100. Poista listasta kaikki lukua 2 isommat luvulla 2 jaolliset luvut. Sitten poista listasta kaikki lukua 3 suuremmat luvulla 3 jaolliset luvut. Jne.

Lisätehtäviä

12) Selvitä, mihin alkulukuja käytetään.

13) Muuta ohjelmaasi niin, että se laskee montako lukua 1000 pienempää alkulukua on olemassa.

14) Muuta ohjelmaasi niin, että se kysyy käyttäjältä alarajan ja ylärajan sekä ilmoittaa kyseisellä välillä olevat alkuluvut

Mahdollisia malliratkaisuja

11)

```
import math

n=100

alkuluvut = []
for i in range(2,n+1):
    alkuluvut.append(i)
i = 2
while(i <= int(math.sqrt(n))):
    for j in range(i*2, n+1, i):
        if j in alkuluvut:
            alkuluvut.remove(j)
    i = i+1
print (alkuluvut)
```

13)

```
n = 1000
lkm = 0
seula = [0]*(n+1)
for i in range(2,n+1):
    for j in range(2*i,n+1,i):
        seula[j] = 1
for i in range(2,n+1):
    if seula[i] == 0:
        lkm += 1
print(lkm)
```

14)

```
alkuMerkkijonona = raw_input("Mista aloitetaan: ");
alku = int(alkuMerkkijonona);
loppuMerkkijonona = raw_input("Mihin lopetetaan: ");
loppu = int(loppuMerkkijonona);
lukumaara=0;
print "Alkulukuja ovat:"
for i in range(alku,loppu+1):
    if i>1:
        for j in range(2,i):
            if (i%j==0):
                break
        else:
            print i
            lukumaara+=1
print "Alkulukuja oli " + str(lukumaara) + " kappaletta."
```

Kilpikonnagrafiikka (Python Turtle), yläkoulu

Anna Leinonen, CC BY-SA 4.0

18.10.2020

Kilpikonnagrafiikkaa

Aihe

Oppituntien (2×45 min) aikana on tarkoitus tutustua Pythonin kilpikonnagrafiikkaan eli Turtle-ohjelmointiin yläkoulun matematiikan tunneilla (7.–9. lk). Tavoitteena on harjoitella erilaisten geometristen muotojen piirtämistä ohjelman avulla, sekä kerrata erilaisia kulmiin liittyviä sääntöjä.

Ennen Turtle-oppitunteja oppilaiden kanssa olisi hyvä olla käyty läpi mitä on ohjelmoinnillinen ajattelu, sekä tutustuttu Python-ohjelmoinnin peruskomentoihin. Myös geometriset muodot ja kulmiin liittyvät säännöt (tasakylkisen ja tasasivuisen kolmion kulmat, sekä vieruskulmat) on hyvä olla hallussa.

Oppimistavoitteet ja arviointi

Oppitunneilla opetelleen repl.it-ohjelmointiympäristössä työskentelyä, opitaan mitä konnagrafiikka on ja miten Pythonilla kirjoitetaan yksinkertaisia ohjauskomentoja konnarobotille. Komentoja käyttämällä opitaan piirtämään erilaisia muotoja.

Kukin oppilas työskentelee omalla tietokoneellaan. Aluksi kukin oppilas luo itselleen repl.it-tilin. Sen jälkeen käymme yhdessä läpi Turtle-kirjaston käyttöönoton ja esimerkkien avulla harjoituksissa tarvittavat käskyt. Tämän jälkeen oppilaat tekevät annettuja harjoituksia omaan tahtiin ja palauttavat tehtävät opettajalle jakamalla linkit (esim. Teams tai Classroom-ryhmässä). Tehtävistä ei anneta pisteitä, vaan oppitunneilla työskentelyä arvioidaan kokonaisvaltaisesti opettajan näkökulmasta ja oppilaan itsearviointiin pohjautuen.

Oppilaat saavat ohjausta ja palautetta opettajalta oppitunnin aikana ja lopuksi tekevät itsearvioinnin omasta työskentelystään.

Työskentelyvälineet ja opetusmenetelmät

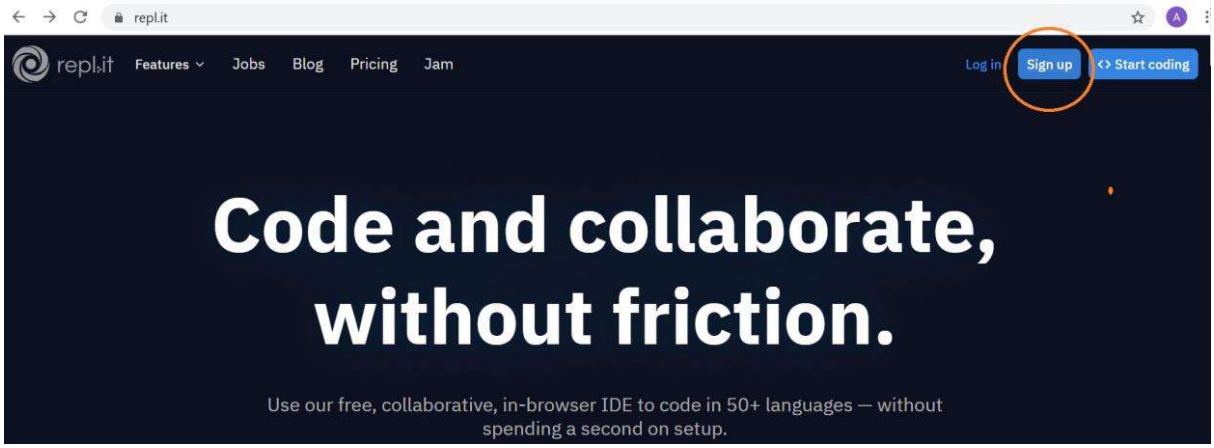
Oppitunneilla koodi kirjoitetaan repl.it-ohjelmontiympäristössä. Näin ollen oppilaan ei tarvitse asentaa mitään erillisiä ohjelmia tietokoneelleen, vaan ohjelmointi tapahtuu tietokoneella internetselaimen kautta.

Koska Turtle-ohjelmointi ei ole luultavasti suurimmalle osalle oppilaista entuudestaan tuttua, on hyvä käydä aluksi perustoiminnot ja käskyt yhdessä läpi opettajajohtoisesti, jonka jälkeen varsinainen oppiminen tapahtuu harjoituksia tekemällä ja kokeilemalla komentoja. Kukin oppilas tekee harjoitukset omalla koneellaan, mutta luokassa on mahdollisuus pohtia ja kokeilla komentoja yhdessä kaverin kanssa. Opettaja on harjoitusten aikana luokassa ohjaamassa työskentelyä.

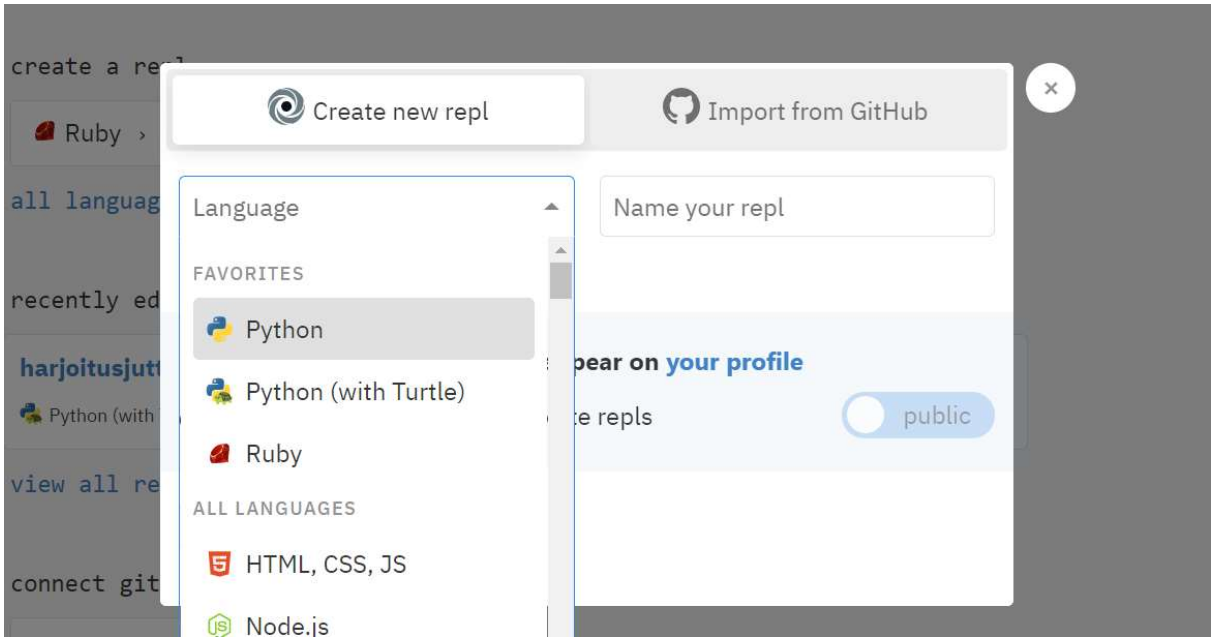
Ohjeet opettajalle ja oppilaille:

Turtle-ohjelmointi repl.it -ohjelmointiympäristössä

- Mene selaimessa osoitteeseen repl.it
- Luo tili kohdasta **sing up**
    - Syötä tarvittavat tiedot ja vahvista tilin käyttöönotto sähköpostiin tulleesta linkistä



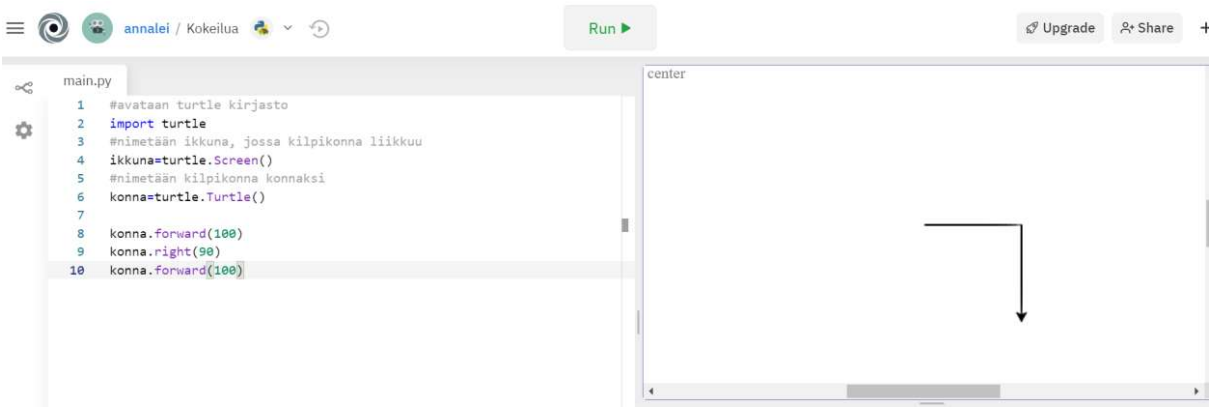
- Kun olet kirjautunut sisään tilillesi, aloita tehtävä valitsemalla:
    - **+New repl**
    - Valitse ohjelmointikieleksi **Python (with Turtle)** ja syötä sopiva nimi tiedostolle (esim. Harjoitus 1)
    - **Create repl**



- Kun halutaan hyödyntää kilpikonnagrafiikkaa Pythonissa, tulee avata turtle-kirjasto, jossa piirtämiseen liittyvät käskyt on määritelty. Kirjaston avaamiseen käytetään **import** käskyä. Lisäksi nimetään ikkuna, jossa kilpikonna liikkuu ja annetaan nimi kilpikonnalle (nimet voi valita itse, mutta ääkkösiä tulee välttää).

- Nyt voi alkaa antaa käskyjä konnalle (korvaa käskyssä konna sana antamallasi kilpikonnan nimellä). Alla olevan taulukon komentoja voi kokeilla alkuun (kirjoita haluamasi komennot komentoikkunaan ja klikkaa run):

import turtle	Turtle-kirjaston käyttöönotto
konna = turtle.Turtle()	Osoittimelle annetaan nimi konna
konna.forward(100)	Konna liikkuu eteenpäin 100 pikseliä
konna.backward(100)	Konna liikkuu taaksepäin 100 pikseliä
konna.right(90)	Konna kääntyy oikealle 90 astetta
konna.left(90)	Konna kääntyy vasemmalle 90 astetta
konna.circle(100)	Piirtää ympyrän, jonka säde on 100



- Lisää komentoja: https://docs.python.org/3/library/turtle.html#overview-of-available-turtle-and-screen-methods

Harjoitus 1:

Piirrä a) neliö, b) kolmio, c) ympyrä

- Oppilaat palauttavat tekemänsä harjoitukset jakamalla linkin opettajalle (Share, invite someone with a join link).

Ominaisuuksia:

- Ennen harjoitusten jatkamista, vielä muutamia ominaisuuksia ja komentoja (lisää komentoja löytyy löytyy aiemmin esitetyn linkin kautta):

ikkuna=turtle.Screen()	Annetaan ruudulle nimi ikkuna
ikkuna.bgcolor("green")	Määrittää ikkunan taustavärin vihreäksi
konna.up()	Kynän voi nostaa, jolloin konna ei piirrä viivaa
konna.down()	Laskee kynän
konna.shape("turtle")	Ohjaimelle voi antaa erilaisia muotoja. Muotoja: arrow, blank,circle, classic, square,triangle, turtle
konna.speed(10)	Määrittääpiirtonopeuden. Parametrit 1-10.
konna.stamp()	Konna leimaa jälkensä ruudulle

konna.setposition(100,100)	Määrittää kynän sijainnin x ja y-koordinaatteina
print(konna.position())	Antaa konnan senhetkisen sijainnin koordinaatteina
konna.color("blue")	Määrittää viivan väriksi sinisen.
konna.fillcolor("red") **konna.begin_fill()** **konna.end_fill()**	Määrittää kuvioiden täyttöväriksi punaisen. Kirjoita juuri ennen kuin alat piirtämään kuviota, jonka haluat värittää. Kirjoita juuri kun olet piirtänyt kuvion.
konna.write("Moi", font=("Arial", 20, "normal"))	Kirjoittaa Moi-tekstin ruutuun.

Harjoitus 2:

Piirrä a) Japanin lippu, b) Suomen lippu, c) keltainen hymiö

Harjoitus 3:

Piirrä ja maalaa mökki, jossa on harjakatto, ikkuna ja ovi. Oven yläpuolella lukee sinun nimesi. Taivaalla paistaa aurinko.

Harjoitus 4:

Piirrä omaa mielikuvitusta käyttäen hieno kuva Turtle-ohjelmalla.

Harjoitus 5:

Kirjoita ohjelma, jossa jokaisesta neljästä nurkasta lähtee liikkeelle kilpikonnat, jotka asettuvat keskellä ruutua päällekkäin.

Harjoitus 6:

Luo kaksi kilpikonnaa (keksi nimet kilpikonnille), joista toinen ensin piirtää vaaleanvihreälle pohjalle mustan neliön ja toinen sen jälkeen pinkin pallon.

- Oppilaat palauttavat kaikki tekemänsä harjoitukset jakamalla linkit opettajalle (Share, invite someone with a join link).

Eriyttäminen:

Oppilaat tekevät omaan tahtiin mahdollisimman monta tehtävää. Nopeimpia ja innostuneimpia oppilaita voi ohjata lisäksi tutkimaan, miten silmukkaa (for) voidaan hyödyntää kuvioita piirrettäessä ja kokeilla esimerkiksi kahdeksankulmion piirtämistä silmukkaa hyödyntäen.

Lähteet:

repl.it

https://peda.net/jyvaskyla/ict/palvelut/ohjelmointi-robotiikka/pop

https://docs.python.org/3/library/turtle.html#overview-of-available-turtle-and-screen-methods

Woodcock Jon: Kivaa koodausta - Opetellaan ohjelmointia, Sanoma Media Finland, 2015

Turtle-harjoitukset:

Harjoitus 1:

Piirrä a) neliö, b) kolmio, c) ympyrä

Harjoitus 2:

Piirrä a) Japanin lippu, b) Suomen lippu, c) keltainen hymiö

Harjoitus 3:

Piirrä ja maalaa mökki, jossa on harjakatto, ikkuna ja ovi. Oven yläpuolella lukee sinun nimesi. Taivaalla paistaa aurinko.

Harjoitus 4:

Piirrä omaa mielikuvitusta käyttäen hieno kuva Turtle-ohjelmalla.

Harjoitus 5:

Kirjoita ohjelma, jossa jokaisesta neljästä nurkasta lähtee liikkeelle kilpikonnat, jotka asettuvat keskellä ruutua päällekkäin.

Harjoitus 6:

Luo kaksi kilpikonnaa (keksi nimet kilpikonnille), joista toinen ensin piirtää vaaleanvihreälle pohjalle mustan neliön ja toinen sen jälkeen pinkin pallon.

Alkulukujen etsintää Python-ohjelmoinnin avulla, yläkoulu

Nimetön,

22.11.2021

Alkulukujen etsimistä Python-ohjelmoinnin avulla

Tuntisuunnitelman tekijä: [redacted]

Suunnittelin kahden oppitunnin mittaisen opetuskokonaisuuden, jossa on tarkoitus laatia Python-ohjelmointikieltä apuna käyttäen algoritmeja alkulukujen etsintään.

Kohderyhmä ja ennakkotiedot

Kokonaisuus on suunniteltu oppilaille, joille Python-ohjelmointikieli on jo entuudestaan tuttua. Tämä opetuskokonaisuus soveltuu parhaiten oppilaille, jotka ovat harjoitelleet Python-ohjelmointikieltä Tie koodariksi –oppimisympäristön avulla. Tässä oppimiskokonaisuudessa käytetään osittain Tie koodariksi –sivuston lukuja 13 ja 14. Näitä lukuja edeltäviin lukuihin tutustuminen on siis oppilaille eduksi, mutta ei välttämätöntä. Nämä oppitunnit voisi pitää soveltavana ohjelmointitehtävänä esimerkiksi 9. luokan keväällä, tai aiemminkin, mikäli oppilailla on Pythonilla ohjelmointi hallussa. Oppitunnit soveltuvat myös lukioon. Opettajalta edellytetään myös Python-ohjelmointikielen hallintaa.

Oppituntien tavoitteet

- matematiikka: oppilas ymmärtää luonnollisen luvun, alkuluvun ja jaollisuuden käsitteet.
- ohjelmointi: algoritmi, jaollisuus, ohjelmointikielen toistorakenne (for –silmukka), lista

Tarvittavat välineet

- Tietokoneet (kaikille oppilaille omat, myös parityöskentelyllä onnistuu)
- Ohjelmointitehtäviä alkuluvuista –tehtävämoniste
- Eratostheneen seula –tehtävämoniste
- Kynät
- Tie koodariksi –sivusto https://tie.koodariksi.fi/
- Tie koodariksi –sivuston editori https://tie.koodariksi.fi/editori
- Nettiyhteys
- Laskimet

Arviointi

- Opettaja arvioi oppilaiden osaamista kiertämällä luokassa ja tarkkailemalla heidän osaamistaan tunnin aikana
- Oppilaiden vertaisarviointi tunnin päätteeksi

Ensimmäinen oppitunti

Ensimmäisen oppitunnin aluksi palautetaan mieleen *alkuluvun* käsite. Opettaja johdattelee oppilaita tunnin teemaan esimerkiksi kyselemällä millaiset luvut ovat alkulukuja. Jos määritelmä ei heti muistu oppilaiden mieleen, voi opettaja johdatukseksi kirjoittaa taululle lukuja ja kysellä, ovatko nämä alkulukuja vai eivät. Näin päästään lopulta palauttamaan mieleen alkuluvun käsite.

Pohditaan seuraavaksi, miten Pythonilla voisi koodata ohjelman, jolla voisi testata, onko jokin luku alkuluku vai ei. Oppilaiden innokkuudesta ja osaamistasosta riippuen oppilaat saattavat löytää tähän itsekin vastauksen, mutta voi olla, että koodin kirjoittamisvaihe mennään aika opettajajohtoisesti. Koodia kirjoitettaessa opettajan on hyvä selittää, miksi mikäkin syntaksi koodiin laitetaan.

Oheisessa kuvassa on esimerkkinä koodi, jolla voi tutkia onko jokin luku alkuluku. Kuvassa on käytetty esimerkkinä lukua 31. Koodin voi kirjoittaa esimerkiksi Tie koodariksi –sivuston editoriin https://tie.koodariksi.fi/editori .

```
1  onAlkuluku = True
2  a = 31
3  for i in range(2,a):
4      if a % i == 0:
5          onAlkuluku = False
6          break
7  print(str(a) + ' on alkuluku ' + str(onAlkuluku))
8
9
10
```

Suorita

```
31 on alkuluku True
```

Kun oppilaat ovat saaneet koodatuksi tämän ohjelman, he voivat tutkia sen toimivuutta erilaisilla luvuilla. Lisäksi he voivat tehdä Ohjelmointitehtäviä alkuluvuista – harjoitusmonisteesta tehtävät 1-4.

Toinen oppitunti

Seuraavalla tunnilla jatketaan työskentelyä alkulukujen ja ohjelmoinnin parissa. Tunnin aluksi pohditaan yhdessä, kuinka alkulukuja voitaisiin systemaattisemmin etsiä kaikkien luonnollisten lukujen joukosta. Yhtenä etsimiskonstina opettaja esittelee Eratostheneen seulan idean. Esittelyn voi tehdä esimerkiksi Wikipedian avulla (https://fi.wikipedia.org/wiki/Eratostheneen_seula) Eratostheneen seulan toimintaperiaatteeseen tutustuttaessa opitaan tai kerrataan *algoritmin* käsite.

Eratostheneen seulan toimintaperiaatteeseen tutustutaan aluksi kynän ja paperin avulla. Mainio valmis pohja löytyy tähän Matikkatunti.fi –siuvstolta (https://drive.google.com/file/d/1GtZxr8N80-FoMROBwCa5JxVGjEXVvyEW/view). Jos tässä vaiheessa oppilaiden motivointi on tarpeen, voi oppilaita haastaa vaikkapa kilpailemaan, kuinka kauan heiltä kuluu aikaa etsiä kaikki alkuluvut ruudukoista. Opettaja voi antaa myös vinkin, että ensimmäisessä ruudukossa alkulukuja on 25 kpl, toisessa ruudukossa 21 kpl ja kolmannessa ruudukossa 16 kpl. Tässä kohdassa oppilaita kannattaa eriyttää: nopeimmat ehtivät etsimään alkulukuja useammastakin ruudukosta, hitaammille riittänee työskentely ensimmäisen ruudukon alkulukujen parissa. Alkulukujen etsintää voi hyvin harjoittaa myös parityöskentelynä. Laskimella voi halutessaan nopeuttaa alkulukujen etsintää.

Tämän jälkeen pohditaan miten koodaamalla voisi tehdä ohjelman, joka noudattaisi Eratostheneen algoritmia. Tähän löytyy opastusta Tie koodariksi –sivuston tehtävästä 14 https://tie.koodariksi.fi/alkeet/14 . Koodia kirjoittaessa opitaan tai kerrataan ohjelmoinnin käsitteet *silmukka, pääsilmukka, sisäsilmukka* ja *lista*. Oheisessa kuvassa on esitettynä koodi ohjelmalle, joka etsii Eratostheneen algoritmia apuna käyttäen kaikki alkuluvut välillä 2-100.

```
n = 100
seula = [0]*(n+1)
for i in range(2,n+1):
    for j in range(2*i,n+1,i):
        seula[j] = 1
    if seula[i] == 0:
        print(i)
```

Suorita

```
2
3
5
7
11
13
17
19
23
```

Kun oppilaat ovat koodanneet tämän ohjelman, he voivat tutkia koodin toimivuutta erilaisilla luvuilla. Lisäksi he voivat tehdä Ohjelmointitehtäviä alkuluvuista – harjoitusmonisteesta tehtävät 5-7.

Oppitunnin päätteeksi oppilaat vaihtavat tehtäväpapereitaan ja tarkastavat tekemänsä tehtävät yhdessä. Jos tarkastusvaiheessa herää kysymyksiä ja keskustelua, niistä voidaan keskustella luokassa yhteisesti.

Lähteet

Eratostheneen seula moniste:

https://www.matikkatunti.fi/index.php/2019/09/13/eratostheneen-seula/

Tie koodariksi –sivusto:

https://tie.koodariksi.fi/

Tie koodariksi –sivuston Python-editori:

https://tie.koodariksi.fi/editori

Tie koodariksi –sivuston tehtävä 14:

https://tie.koodariksi.fi/alkeet/14

Muut:

https://fi.m.wikipedia.org/wiki/Eratostheneen_seula?wprov=sfla1

http://www.hbmeyer.de/eratclass.htm

https://maol.fi/materiaalit/kpm/7-luokka/p1/2ljl/2-8-alkuluvut/

Ohjelmointitehtäviä alkuluvuista

Muista! *Alkuluku* on lukua 1 suurempi luonnollinen luku, joka ei ole jaollinen muilla positiivisilla kokonaisluvuilla kuin itsellään ja luvulla 1.

Ratkaise seuraavat tehtävät koodaamalla aluksi ohjelma jonka avulla voit tutkia, onko jokin luku alkuluku.

1. Onko luku 31 alkuluku?

2. Etsi kaikki alkuluvut (8 kpl) väliltä 2-20.

3. Etsi viisi alkulukua väliltä 40-100.

4. Ympyröi ruudukosta kaikki alkuluvut.

41	43	44	47
91	93	97	99
171	173	174	177
211	213	217	219

Ratkaise seuraavat tehtävät koodaamalla aluksi ohjelma jonka avulla voit etsiä alkulukuja hyödyntäen Eratostheneen seulan algoritmia.

5. Luettele kaikki alkuluvut (6 kpl), jotka löytyvät väliltä 200-250.

6. Mikä on alkuluku, joka on lähinnä lukua 2000?

7. Mikä on suurin alkuluku, jonka onnistuit löytämään laatimasi ohjelman avulla?

Liikunnallista ohjelmointia, yläkoulu

1.4.2022

Tuntisuunnitelma ohjelmoinnin opettaminen

Oppiaineista mukana ovat matematiikka ja liikunta, avaruudellinen hahmottaminen, ohjelmointi, asteluku sekä suunnistus. Oppilaani eivät ole koskaan käyttäneet ohjelmointiin mitään ohjelmaa, joten aloitamme nollasta. Rikkonaisen koulunkäyntihistorian vuoksi näin ysiluokan keväällä täytyy moni asia käydä läpi vauhdilla. Tavoitteena on ymmärtää algoritmista ajattelua sekä tutustua visuaaliseen ohjelmointikieleen.

Orientaatio ja motivointi: Opettaja seisoo silmät lähes sidottuina luokan ovella. Opettaja pyytää oppilaita asettumaan riviin ja jokainen oppilas saa vuorotellen sanoa opettajalle ohjeen, jota opettaja noudattaa. Tavoitteena on saada opettaja kulkemaan melko turvallisesti ovelta pöytänsä luo. Opettaja ei ymmärrä epämääräisiä ohjeita ja voi tarvittaessa tehdä tarkentavia kysymyksiä.

Tehtävä 1: Meillä on koulussa asfalttiin alakoulun puolelle maalattu satataulu (jos satataulua ei ole, aulatila tai liikuntasali sekä maalarinteippi ovat oivaksi avuksi). Oppilaiden on suunniteltava ruutupaperille tai satataulupaperille labyrintti sekä ohjeistus parilleen sen läpäisemiseksi. Kertaamme aluksi komennot (askelten lukumäärä, käänny 90 astetta suuntaan x, kertaamme oikean ja vasemman).

Jokainen saa parinsa kanssa testata suunniteltua labyrinttia ulkona. Labyrintissa kulkija noudattaa annettuja ohjeita ja piirtää katuliidulla reittinsä maahan tai suoraan ruutupaperille (kulkija voi piirtää reitin ruutupaperille myös harjoituksen lopuksi). Pari tarkistaa, pääsikö kaveri ulos oikeaa reittiä. Kulkija arvioi komentojen oikeellisuutta ja labyrintin komentojen laatija itsearvioi, mitkä asiat onnistuivat ja mitä tekisi toisin.

(Eriyttävä tehtävä ohjelmoinnin harjoitteluun, omille oppilailleni haastava)

Tehtävä 2: Kun kaikki ovat saaneet testata labyrinttejaan, luodaan oppilaille tunnukset Scratchiin. Käydään yhdessä läpi, mikä Scratch on ja mitä komentoja sieltä löytyy, perusperiaatteet mm. silmukasta. Suomennetaan ja avataan toimintojen värit taululle. Oppilaat toteuttavat (vahvasti tuettuna ja sovellettuna esim. ei aarretta eikä vihollista) materiaalipankista Labyrintti-tehtävän https://www.cs.helsinki.fi/group/linkki/materiaali/lapsiohjeet/labyrintti/labyrintti.pdf. Tai jopa ensimmäisen keräilypelin, siinäkin riittää tekemistä.

Oppimiskokonaisuuden tehtävä 1 vaatii 1-2 x 45 min. riippuen tietysti labyrinttikokeilun ruudukkojen määrästä. Jos toteuttaa myös ohjelmointitehtävän eli tehtävän 2 tietokoneella, täytyy lisäksi varata vähintään 3 x 45 minuutin oppituntia. Kokonaisuuteen menee siis 5 x 45 minuuttia. Jos toteutetaan Scratch-osio, oppilaat mieluusti testaavat toistensa pelejä sekä antavat niistä toisilleen palautetta.

Tarvitaan ruutupaperit, satataulut paperille, kynät, (maalarinteippiä ja mitta), tietokoneet. Lisätehtävävaiheessa tuodaan vahvasti esiin ohjelmoinnin jakamisen periaatteita ja toisten koodien uudelleenkäyttöä, eli rohkaistaan auttamaan kaveria. Verrataan tätä jakamisen kulttuuria heille tuttuihin tiktok-editeihin.

Sovelluksen ohjelmointi code.org:n App Labissa, yläkoulu/lukio

Suula Arppe, CC BY-SA 4.0

6.1.2020

Tuntisuunnitelma, 2 oppituntia, 90 min

Aihe: Sovellusten teko

Tuntisuunnitelma on yläkouluun tai lukioon matematiikan opetukseen. Aikaisempaa ohejlmointikokemusta ei tarvita. Sovellukseen voidaan edellyttää jonkin tietyn aihealueen käyttämistä esimerkiksi geometrisia kuvioita tai koordinaatistoa. Sovelluksen tekeminen voidaan yhdistää ilmiöopiskeluun ja sitä kautta hyödyntää myös muita oppiaineita.

Oman sovelluksen teko vaatii ohjelmoinnin lisäksi myös mm. matematiikan, englannin ja kuvataiteen osaamista. Projekti voidaan toteuttaa joko 2 oppitunnin pituisena tai laajempana työnä ilmiöopetuksen yhteydessä. Ilmiöopetuksen myötä sovelluksille voidaan antaa jokin tietty aihe.

Oppimistavoitteiden määrittely ja arviointi

Tavoitteena on tutustua ohjelmointiin ja matematiikan hyödyntämiseen sovellusten tekemisessä. Tavoitteena on myös oppia ryhmätyöskentelytaitoja ja tutustua sovellusten tuottamisen eri vaiheisiin.

Opiskelijat tuottavat yhdessä arviointikriteerit, miettimällä mikä tekee sovelluksesta hyvän. Opiskelijoiden keksimät kriteerit kerätään esim. taululle ja niiden pohjalta muut antavat vertaisarviointia sekä sanallisesti että arvosanalla. Jokainen opiskelija itsearvioi sekä oman työnsä sovittujen kriteerien avulla että oman panoksensa ryhmän toimintaan. Opiskelija arvioi myös omaa kehittymistään (tähän voi tehdä tunnin alkuun itsearvioinnin ja tunnin lopussa katsotaan, miten eri kohdissa on kehittynyt).

Sovelluksen arvioinnin lisäksi opettaja arvioi formaalisesti opiskelijoiden toimintaa ja varmistaa myös valitun matematiikan aihealueen näkymisen työssä. Projekti arvioidaan kuitenkin kokonaisuutena osana kurssiarviointia, opettaja voi ottaa huomioon oman arviointinsa lisäksi myös itsearvioinnin ja vertaisarvioinnin.

Työskentelyvälineet ja opetusmenetelmät

Välineet: Tietokone vähintään yksi per ryhmä.Sovellus tehdään selaimessa https://code.org/educate/applab-sivustolla. Sivusto vaatii kirjautumisen, joten voi olla järkevää luoda väliaikaiset tilit opiskelijoille tai käyttää koulun sähköposteja. Opettaja voi tehdä omat tunnukset sivustolle ja luoda oman kurssin, jonne opiskelijat voivat liittyä koodilla. Opettajan kurssin avulla voi mahdollistaa oppilaiden samanaikaisen projektin muokkauksen.

Tunnin kulku
1. Opiskelijat jaetaan 2-3 hengen ryhmiin. Opiskelijoita pyydetään miettimään, millaisia sovelluksia he käyttävät joka päivä. Opiskelijoita pyydetään myös miettimään, mikä tekee sovelluksesta hyvän. Heidän tulee kerätä 3-4 tärkeintä kriteeriä hyvälle sovellukselle. Kriteerit kerätään yhdessä esim. taululle.

2. Code.org-sivustolta löytyy tutoriaali, jonka opiskelijat katsovat.
https://studio.code.org/s/applab-intro/stage/1/puzzle/1

3. Ryhmät miettivät, millaisen sovelluksen he haluaisivat tehdä. He voivat katsoa ideoita sivustolta löytyvistä malleista. Kun he ovat päättäneet sovelluksen tyypin, heidän tulee miettiä paperille, miten sovellus toimii. Kun heillä on idea sovelluksen toiminnasta, alkavat he tehdä sovellusta Sovelluslabran avulla. Tehtävää voi eriyttää melko helposti, sillä sovellusta voi koodata joko visuaalisesti erilaisilla lohkoilla tai tekstipohjaisena. Ryhmiä kannustetaan tekemään yhteistyötä: jos ryhmä jää jumiin, kannustetaan heitä kysymään muilta ryhmiltä apua.

4. Ryhmiä kannustetaan testaamaan sovellustaan muilla ja pyytämään palautetta. Saadun tiedon avulla he voivat vielä muokata sovellustaan toimivammaksi.

4. Projektit jaetaan yhteiseen dokumenttiin ja jokainen oppilas käy kokeilemassa jonkin toisen ryhmän tuotosta. Oppilaat antavat aiemmin mietittyjen hyvän sovelluksen kriteerien pohjalta vertaisarviointia ja tekevät myös itsearvioinnin. Myös opettaja antaa formatiivista palautetta ja arvioin yhteisten kriteerien mukaisesti sovellukset.

Todennäköisyyksien laskentaa Pythonilla, yläkoulu/lukio

Annukka Rautiola, CC BY-SA 4.0

11.6.2020

Algoritmitnen ajattelu - oma tuntisuunnitelma

Todennäköisyyksien laskentaa Pythonilla

Kohderyhmä: Peruskoulun 9. Luokkalaiset, joilla todennäköisyyslaskenta on ajankohtaista. Toimii varmaankin myös lukion ensimmäisinä ohjelmointitehtävänä tai peruskoulun muilla yläluokilla.

Välineet: Oma tietokone tai chromebook, tablettikin voi olla välttävä, vaikkei paras mahdollinen laite. Käytetään verkkoselaimessa toimivaa repl.it - alustaa. Alustalle ei tarvitse kirjautua/ luoda tunnuksia, jos tehtävän yhden vaiheen pystyy saattamaan loppuun oppitunnin aikana. Kirjautuminen on kuitenkin yksinkertaista esim. Olemassaolevilla oppilas(google)-tunnuksilla.

Esitiedot / taidot: Tätä harjoitusta varten on hyvä tuntea joitakin Python -komentoja etukäteen:

- Turtle -toiminnot
- Silmukka
- Ehtolause
- Muuttuja

Riittävät taidot saa esim. tie.koodariksi -sivuston ohjelmointitehtäviä tekemällä. Tätä harjoitusta edeltäväksi työksi sopisi myös hyvin Koodiaapisen tekijöiden taideharjoite, jossa kilpikonnatoiminnot tulisivat tutuiksi.
https://docs.google.com/document/d/1Jhuqlmji5N0gXHr6cttjv5rp1anfXlJLXzLkbdnAzyo/pub

Tavoitteet: Tuoda ohjelmionti osaksi matematiikan opetusta molempia osuuksia hyödyttävällä tavalla. Oppilaan tulisi hahmottaa todeenäköisyyslaskennan perusidea - se, että yksittäisen nopanheiton tulosta on vaikea ennustaa, mutta mitä useampia heittoja on, sitä enemmän niiden tulokset noudattavat klassisen todennäköisyyden kaavaa. Oppilas lisäksi huomaa, että käytännössä hankalasti toteutettavat suuret heittomäärät saadaan ohjemoinnilla näkyviin melko nopeasti.

Arviointi: Oppilaan itsearviointi sekä kehu kaveria - arvio palautuksen yhteydessä kirjallisesti. Opettaja arvioi työskentelyä ja parjaansa yrittämistä ja antaa palautetta suullisesti tuntien aikana tai wilma-merkinnän muodossa.

Alustus:
Tutustutaan klassiseen todennäköisyyteen:
“Olkoon satunnaisilmiöllä n yhtä todennäköistä tulosvaihtoehtoa ja tarkastellaan ilmiön tapahtumaa, johon liittyy tulosvaihtoehdoista k kpl, joita sanotaan ko. tapahtumalle suotuisiksi. Tällöin ko. tapahtuman klassinen todennäköisyys p on ko. tapahtumalle suotuisien tulosvaihtoehtojen osuus satunnais-ilmiön kaikista mahdollisista tulosvaihtoehdoista.”
Ilkka Mellin, TKK: Todennäköisyyslaskenta: todennäköisyyslaskenta ja sen laskusäännöt
http://math.tkk.fi/opetus/sovtoda/oppikirja/TodLaskLaskusaannot.pdf

Käydään läpi **esimerkki**: heittäessä 20-sivuista noppaa on todennäköisyys sille, että saadaan numero 3 on 1/20. Oppilaat selvittävät ryhmittäin todennäköisyyden seuraaville tapahtumille:

a) Kolikonheitossa klaava
b) Tavallisessa nopanheitossa numero 5
c) Tavallisessa nopanheitossa jokin numeroista 1-4
d) Satunnaisesti valitulla luokkatoverilla on poninhäntä

Katsotaan Samun tiedekanavan video: Nopanheitto ja todennäköisyyden luonne
https://www.youtube.com/watch?v=p1URCLkTLNA

Varsinainen ohjemointiosuus: **Koodataan itse samanlainen todennäköisyyskaavio kuin Samulla - voimme hahmottaa useiden noppien heittojen tuloksia!**

Oppilaat saavat linkkinä jaettuna valmiin koodinpätkän, jota tulee muuntaa toivottuun suuntaan. Koodin kommenteissa on ohjeet työskentelyyn. Oppilaat voivat keskenään keskustella vaihtoehdoista ja pyytää apua.

Vaihe 1:

- Kaavion piirtäminen. Oppilaan tulee muokata koodia siten, että kilpikonnatoiminnoilla saadaan halutunlainen kasa suorakulmioita, mm. Säätää korkeutta, kääntymiskulmaa, sopivaa siirtymistä uuteen aloituspisteeseen, toistojen määrää.
- Oppilasta ohjataan kokeilemaan koodinmuokkausta vaiheittain “Run”-toiminnolla
- Oppilas luo docs- tai muun vastaavan tiedoston, johon liittää kuvankaappauksen muokkaamastaan koodista (ota useita, jollei yhteen kaappaukseen mahdu koko koodi).

https://repl.it/@AnnukkaRautiola/Nopanheittosimulaattori-vaihe-1-1
(kuva koodista liitteenä)

Kuvio näyttää tältä, kun oppilas onnistuu piirtämään yhden laatikon. Vielä tarvitaan toisto ja siirto uuteen aloituspisteeseen.

Malliesimerkki tämän vaiheen onnistuneesta ratkaisusta
https://repl.it/@AnnukkaRautiola/Nopanheittosimulaattori-malliesimerkki-vaiheeseen-1

- Keskustelu / havainnot:
  - Kaavion laatikot ovat kaikki yhtä suuria
  - Satunnaisuudella ei ole mitään tekemistä vielä tämän version kanssa
  - Ei simuloi nopanheittoa → siirrytään vaiheeseen 2

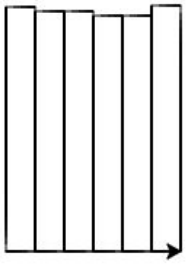
Vaihe 2:

Uusi linkki koodiin, joka on jo valmiimpi.

- Oppilaan tulee muokata nopanheiton silmälukuja, luoda riittävästi muuttujia ja keksiä, miten kaavion piirto-ohjeita tulee muuntaa, jotta lopputulos mahtuisi ruutunäkymään.
- Kokeile erilaisilla heittomäärillä silmälukujen käyttäytymistä.
- Lisää kuvankaappaukset tästäkin koodista ja valmiista nopanheittokaaviosta, liitä ne aiempaan tiedostoon.

https://repl.it/@AnnukkaRautiola/Nopanheittosimulaattori-vaihe-2

Vaihekuvat

Heittoja 10	Heittoja 100 000
		
Nopanheitto onnistuu, mutta pylväiden kokoa pitää vielä skaalata.	Nyt näyttää jo paremmalta! Nopanheittoja on kuitenkin vielä melko vähän, jos kaavio näyttää tältä.	Riittävästi nopanheittoja (ja pylväiden korkeus skaalattu siten, että kuvio mahtuu ruudulle) → Simulaattori on valmis!

Lopputarkastelu ja keskustelu:

- Lopuksi käydään yhdessä läpi esimerkkilopputulos joko jonkun oppilaan luomasta simulaattorista tai opettajan esimerkkikoodista.
- Keskustellaan raa'an voiman mahdista: suurissa heittomäärissä tietokoneen laskentateho on yliviomainen.

https://repl.it/@AnnukkaRautiola/Valmis-nopanheittosimulaattori

Vaihe 3. Nopeille ja näppärille (tai lisänä yhteiseen keskusteluun)

- Niille oppilaille, jotka saavat edelliset osiot tehtyä hujauksessa
- Muunna koodia:
    - Mitä jos noppa olisikin tavallisesta 6-tahkoisesta poikkeava?
    - Voisiko simuloida vaikkapa kolikonheittoa / korttipelin todennäköisyyksiä?
    - Saatko piirrettyä kaavion, jonka pylväät ovat erillillään toisistaan?
    - Saatko väritettyä kaavion pylväät? (etsi googlettamalla turtle-komentoja, jotka sopisivat tähän)

Itsearviointi - liitetään osaksi palautettavaa tiedostoa.

1) Miltä ohjelmointitehtävä tuntui? 1 (vaikealta) - 5 (helpolta)
2) Ymmärsin, mitä koodin eri osat tekevät. 1(en lainkaan) - 5(erinomaisesti), perustele
3) Sain apua, millaista ja keneltä? 1(en lainkaan) - 5(jatkuvasti), perustele)
4) Auttoiko ohjelmointitehtävä todennäköisyyslaskennan hahmottamisessa? Mitä hyötyä siitä voisi olla?
5) Anna sanallisesti positiivista palautetta jollekin luokkatoverillesi, joka teki hyvää työtä, auttoi sinua tai oivalsi jotain tuntien aikana. Kerro tähän kenelle ja millaista palautetta annoit.

LIITE: LINKKIEN TAKAA LÖYTYVÄT KOODIT JA NIIHIN UPOTETUT OHJEISTUKSET.

VAIHEEN 1 OHJEISTUS:

```
# Näillä komennoilla haetaan kirjastosta käyttöön tarvittavat toiminnot
from random import *
import turtle
turtle.penup()
# Tässä määrätään kilpikonna siirtymään sopivaan pisteeseen piirtämisen aloitusta varten.
# Muokkaa pisteen koordinaatit siten, että kilpikonna näkyy piirtonäkymässä. Kokeile vaiheittain
"run-toiminnolla" ruudun yläreunassa.
turtle.goto(-100,-100)
# Tämä luo silmukan, jonka toistomäärät määrätään suluissa.
# Tämän alle sisennettynä kirjoitetut toiminnot toistetaan annetun määrän verran.
# Muokkaa toiston määärä sopivaksi vasta kun olet saanut alemman kuvionpiirtotehtävän tehtyä.
for i in range(0,6):
 #Laitetaan kynä alas - tämän jälkeen liikkumisista jää jälki näkymään eli piirretään.
 turtle.pendown()
 #Piirrä nopanheittokaavion yksi osa. Määrää piirroksen muoto ja koko lisäämällä sulkuihin sopivia
lukemia etenemiseen ja kääntymiseen (asteina). Käytä näitä toimintoja- voit kirjoittaa lisärivejä
tarpeen mukaan.
 turtle.forward(10)
 turtle.left(90)
 #Tässä määrätään kilppari siirtymään seuraavan särmiön aloituspisteeseen piirtämättä viivaa
matkalla
  turtle.penup
 turtle.left(0)
 turtle.forward(0)
```

MALLIESIMERKKI VAIHEESEEN 1:

```
import turtle
from random import *
for i in range(0,6):

 turtle.forward(20)
 turtle.left(90)
 turtle.forward(40)
 turtle.left(90)
 turtle.forward(20)
 turtle.left(90)
 turtle.forward(40)

 turtle.penup
 turtle.left(90)
 turtle.forward(20)
```

VAIHEEN 2 OHJEISTUS:

```
#Nämä tuovat ohjelmaan tarvittavat toiminnot kirjastosta.
from random import *
import turtle
# Luodaan vakiot a-? laskemaan saatujen silmälukujen määrää. Vakio a laskee silmälukuja 1.
# Luo riittävän monta vakiota. Aseta alkuarvoksi 0, sillä noppaa ei ole vielä heitetty kertaakaan
eikä yhtäkään silmälukua saatu.
```

```
a = 0
# Tämän silmukan toistomäärä määritellään suluissa. Silmukka käy läpi jokaisen kokonaisluvun
annetulta väliltä.
# Nopanheittojen lukumäärä määritetään tällä
for j in range (0,5):
 # Muuttuja i saa jokaisella nopanheitolla arvon annetulta väliltä
 # Määrää nopanheittoon sopivat arvot sulkuihin ts. mitä lukuja nopalla on mahdollista saada?
 i= randint(1,2)
# Tämä pätkä on ehtolause silmukan sisällä. Jos nopanheitolla vakio i saa arvon 1, vakion a arvo
kasvaa yhdellä. Vakio a siis pitää kirjaa nopanheiton silmälukujen määrästä.
# Jatka muodostamalla ehdot myös muiden silmälukujen määrän laskemiseksi.
 if i == 1:
   a += 1

#Tämä kohta määrää aloituspisteen kilpikonnalle
turtle.penup()
turtle.goto(-60,0)
# Tässä muodostetaan lista, jolle lisätään muuttujat a-f. Tämän ansiosta laskettuja silmälukumääriä
voidaan käyttää myöhemmin kaavion muodostamisessa.
# Lisää listaan luomasi muuttujat.
lista = []
lista.append(a)
# Tässä tehdään silmukka, joka käy läpi listan jäsenet eli alkiot. Ensimmäinen jäsen on
järjestysnumerolla 0.
# Etsi sopivat rajat lukuvälille siten, että kaaviossa käytetään kaikkia listasi muuttujia.
for i in range(0,1):
 turtle.pendown()
 # Kilpikonna piirtää särmiön, jonka korkeus riippuu muuttujista a-? listan mukaisessa
järjestyksessä.
 # Muuta suluissa olevat arvot siten, että suurilla nopanheitoilla saadaan ruudulle mahtuva kaavio.
Voit käyttää eri laskutoimituksia suluissa eteenpäinliikkumisen säätelemiseksi.
 turtle.forward(20)
 turtle.left(90)
 turtle.forward(lista[i])
 turtle.left(90)
 turtle.forward(20)
 turtle.left(90)
 turtle.forward(lista[i])
 #Nostetaan kynä ja siirretään kilppari seuraavan laatikon aloituspisteelle
 turtle.penup()
 turtle.left(90)
 turtle.forward(20)
```

VALMIIN SIMULAATTORIN ESIMERKKIKOODI:

```
from random import *
import turtle

a = 0
b = 0
c = 0
d = 0
e = 0
f = 0

for j in range (0,10):
 i= randint(1,6)

 if i == 1:
   a += 1
 if i == 2:
   b += 1
 if i== 3:
   c += 1
```

```
 if i== 4:
   d += 1
 if i== 5:
   e += 1
 if i== 6:
   f += 1
turtle.penup()
turtle.goto(-60,0)

lista = []
lista.append(a)
lista.append(b)
lista.append(c)
lista.append(d)
lista.append(e)
lista.append(f)

for i in range(0,6):
 turtle.pendown()
 turtle.forward(20)
 turtle.left(90)
 turtle.forward(lista[i]*30)
 turtle.left(90)
 turtle.forward(20)
 turtle.left(90)
 turtle.forward(lista[i]*30)
 turtle.penup
 turtle.left(90)
 turtle.forward(20)
```

Matka-aikojen laskemista Pythonilla, yläkoulu/lukio

Katariina Piri,

14.4.2021

Yläkoulu/lukio: Lasketaan matka-aikoja käyttäen python3-kieltä

Hyödynnetään fysiikassa opittuja matkan ja ajan suureita sekä niiden johdannaista suuretta nopeutta.

Tehtävän voi teettää jokaisen oppilaan itsenäisenä tehtävänä tai 2-3 oppilaan ryhmissä. Opettaja ohjaa työskentelyä ja innostaa yhteiseen pohdintaan/kokoaa keskeiset huomiot.

Esitiedot

Kuva esim. laskukolmiosta ja suureiden suhteista toisiinsa sekä suureiden yksiköistä: matka (s) – km, aika (t) – h, nopeus (v) – km/h

Työvälineet

Python3 -ympäristö tai web-sivusto, esim. replit.com. Kuvaajan piirtämiseksi käytetään python-pakettia matplotlib, joka on käytettävissä myös replit-palvelussa rekisteröidyttyä.

Oppimistavoitteet ja arviointi

Opitaan miten nopeus vaikuttaa matka-aikaan, ja havaitaan miten helppoa ohjelmallisesti on vertailla tietoja (eli oppilas ymmärtää noiden suureiden suhteet ja kuinka saadaan helposti generoitua data josta voidaan vetää johtopäätöksiä).

Oppilas osaa arvioida ohjelman avulla kuinka kauan hänellä menee aikaa esimerkiksi koulumatkaan riippuen siitä kuinka nopeasti liikkuu, esim. kävellen, juosten tai polkupyörällä. Lisäksi maantienopeuksia ja -matkoja vertaillessa pystyy päättelemään kuinka suuri ajallinen hyöty saavutetaan ajamalla esimerkiksi ylinopeutta tai valitsemalla eri kulkuvälineitä.

Ajattelin itse, että tässä harjoituksessa ei arvioitaisi itse ohjelmointia vaan arviointikriteerinä olisi se, että oppilas osaa arvoja muuttamalla määrittää esimerkiksi ajan mikä menee eri kulkutavalla/nopeudella itselleen tärkeän paikkaan. Tai yhtä hyvin kysymyksenä voi olla jokin tietty matka johonkin nimettyyn paikkaan eri kulkuneuvoilla mentynä (esim. toinen ajaa maata pitkin moottoripyörällä 5x pidemmän matkan ja toinen menee suoraan lahden poikki veneellä hitaammin).

Toki etenkin lukiossa lisätehtävänä (tai varsinaisena tehtävänä edistyneemmille) voi antaa esimerkiksi ilmanvastuskertoimen ja eri kulkuvälineiden polttoaineen kulutuslukemat ja pyytää oppilaita ideoimaan, miten sen koodaisi tuohon/lisäisi koodin. Sen onnistumista voisi arvioida niin opettaja kuin etenkin toiset oppilaat vertaisarviointina (esim. oppilaat voisivat testata ja arvioida pareittain koodia).

Esitiedot ja pikakertaus

Aiemmin on tutustuttu python-kieleen, syötteen lukemiseen ja muuntamiseen kokonaisluvuksi, sekä for-silmukoihin ja range -funktioon. Fysiikassa on opittu keskinopeus, v = s / t.

Ohjelmointi

Pyydetään käyttäjältä alussa tiedot, joilla suoritetaan laskutoimitukset.

```
print("Syötä seuraavat tiedot:")
matka = int(input("matka: "))
nopeus_min = int(input("pienin nopeus: "))
nopeus_max = int(input("suurin nopeus: "))
nopeus_askel = int(input("nopeuseron koko: "))
```

Luodaan silmukka, jonka avulla käydään läpi laskutoimitus valituille arvoille

```
for nopeus in range(nopeus_min, nopeus_max+1, nopeus_askel):
    print("nopeus: {} km/h, aika: {} min".format(nopeus, 60 * matka // nopeus))
```

Valmis ohjelma:

```
print("Syötä seuraavat tiedot:")
matka = int(input("matka: "))
nopeus_min = int(input("pienin nopeus: "))
nopeus_max = int(input("suurin nopeus: "))
nopeus_askel = int(input("nopeuseron koko: "))

for nopeus in range(nopeus_min, nopeus_max+1, nopeus_askel):
    print("nopeus: {} km/h, aika: {} min".format(nopeus, 60 * matka // nopeus))
```

Tuloste:

```
Syötä seuraavat tiedot:
matka: 50
pienin nopeus: 10
suurin nopeus: 140
nopeuseron koko: 10
nopeus: 10 km/h, aika: 300 min
nopeus: 20 km/h, aika: 150 min
nopeus: 30 km/h, aika: 100 min
nopeus: 40 km/h, aika: 75 min
nopeus: 50 km/h, aika: 60 min
nopeus: 60 km/h, aika: 50 min
nopeus: 70 km/h, aika: 42 min
nopeus: 80 km/h, aika: 37 min
nopeus: 90 km/h, aika: 33 min
nopeus: 100 km/h, aika: 30 min
nopeus: 110 km/h, aika: 27 min
nopeus: 120 km/h, aika: 25 min
nopeus: 130 km/h, aika: 23 min
```

Voidaan syöttää erilaisia arvoja ja pohtia miten nopeus vaikuttaa matka-aikaan.

Kuvaajien lisääminen ohjelmaan

Laajennetaan ohjelmaa, ja lisätään kuvaaja käyttäen matplotlib.pyplot -kirjastoa.

Tuodaan kirjasto ohjelman alussa ja annetaan sille lyhyempi nimi ohjelmassa:

```
import matplotlib.pyplot as plt
```

Luodaan lista arvoista x- ja y-akseleita varten. x-akselille tulee matkan arvot ja y-akselille matkaan kulutettu aika silmukan iteraatiossa olevalla *nopeudella*. Lisätään kaavioon kuvaaja x- ja y-arvoilla sekä nimetään kuvaaja nopeuden mukaan:

```
x = list(range(matka))
y = [60 * y / nopeus for y in x]
plt.plot(x, y, label="{} km/h".format(nopeus))
```

Silmukan jälkeen lisätään x- ja y-akseleille kuvaukset, otsikko kaaviolle, sekä kuvaajien määritteet.

Lopuksi näytetään kaavio:

```
plt.xlabel("matka, km")
plt.ylabel("aika, min")
plt.title("Matka-ajat {} km matkalla nopeuksilla {}-{} km/h".format(matka, nopeus_min, nopeus_max))
plt.legend()
plt.show()
```

Valmis ohjelma:

```
import matplotlib.pyplot as plt

print("Syötä seuraavat tiedot:")
matka = int(input("matka: "))
nopeus_min = int(input("pienin nopeus: "))
nopeus_max = int(input("suurin nopeus: "))
nopeus_askel = int(input("nopeuseron koko: "))

for nopeus in range(nopeus_min, nopeus_max+1, nopeus_askel):
   print("nopeus: {} km/h, aika: {} min".format(nopeus, 60 * matka // nopeus))
   x = list(range(matka))
   y = [60 * y / nopeus for y in x]
   plt.plot(x, y, label="{} km/h".format(nopeus))

plt.xlabel("matka, km")
plt.ylabel("aika, min")
plt.title("Matka-ajat {} km matkalla nopeuksilla {}-{} km/h".format(matka, nopeus_min, nopeus_max))
plt.legend()
plt.show()
```

Lisäpuuhaa

- Suureiden vaihtaminen eri akseleille ja laskutoimituksien vaihto.
- Liukulukujen syöttö ohjelmaan.
- Miten ohjelman saa hajotettua syötteellä, ja millä tavoin korjata se hyväksymään hajottava syöte.

Pitkän matematiikan todennäköisyyslaskentaa ohjelmoimalla, lukio

6.1.2020

Tuntisuunnitelma

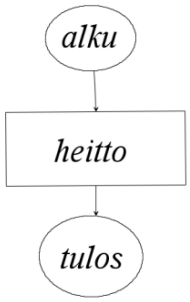
Tuntisuunnitelma on suunnattu MAA10-kurssille eli pitkän matematiikan todennäköisyyslaskentaan. Harjoitus on tarkoitus tehdä kurssin alkupuolella, mutta ei kuitenkaan ihan ensimmäisellä tunnilla. Opiskelijoilla tulee olla todennäköisyyden peruskäsitteet jo vähän tuttuja.

Valitsin harjoitukseen todennäköisyyskurssin, koska kirjoittamalla ohjelmia on helppo tehdä toistoja ja tutkia, minkälaisia tuloksia tulee, kun esimerkiksi noppaa heitetään 1000 tai vaikka 1 000 000 kertaa. Tällaiset kysymykset ovat tärkeitä kurssilla, mutta kukaan ei kuitenkaan voi alkaa näitä testejä tekemään itse käsin.

Alkuasetelma: Tämä harjoitus toimii paremmin, mitä pienempi opetusryhmä on kyseessä. En itse lähtisi kokeilemaan sitä esimerkiksi 35 opiskelijat ryhmälle. Sanoisin, että noin 20 opiskelijaa on maksimi. Harjoitus voidaan ja ehkä kannattaakin tehdä pareittain. Ennen sen tekoa voisi opiskelijoilta tiedustella, minkälaiset ohjelmointitaidot heillä on etukäteen ja yhdistää joko aloittelija – kokeneempi parit tai mielenkiintoista voisi olla myös kokeilla aloittelija – aloittelija ja kokenut – kokenut paritus.

Tuntisuunnitelmassa olen ajatellut, että ohjelmointi tehdään Javalla, mutta yhtä hyvin se voidaan tehdä myös Pythonilla, jos joku opiskelija on kokeneempi käyttämään sitä. Opiskelijoiden tulee ennen tuntia ladata itselleen Java ja Netbeans. Latauksiin löytyy selkeät ohjeet esimerkiksi tällä kurssillakin useaan otteeseen mainitusta ohjelmoinnin MOOCista. (https://ohjelmointi-20.mooc.fi/) Harjoituksessa on myös oletus, että opettajalla on Javan perusteet hallussa. Olio-ohjelmointia ei tarvitse osata, mutta silmukat ja olioviittaukset kylläkin.

Suunnitelma: Tunti aloitetaan miettimällä kolikon heittoa ja kirjoitetaan aluksi ohjelma, joka heittää kolikkoa. Opiskelijat voivat vuokaavion avulla miettiä, miten ohjelman tulisi toimia. Ihan ensimmäinen vuokaavio voi olla mahdollisimman yksinkertainen, esimerkiksi vain:



Tämän jälkeen voidaan miettiä, mitä eri vaiheisiin tarvitaan. Opiskelijoille voi antaa valmiin pohjan koodin kirjoittamista varten, joten heidän ei tarvitse itse miettiä esimerkiksi main metodin tekemistä. Annettu pohja voisi olla esimerkiksi:

```
public class todennakoisyys {
    public static void main(String[] args) {

        //kirjoita oma koodi tähän
    }

}
```

Aluksi opiskelijat kirjoittavat siis ohjelman, jossa kolikkoa heitetään vain kerran. Tähän tarkoitukseen hyvä työkalu on Math-luokan random-metodi, joka palauttaa satunnaisluvun väliltä 0 – 1 (nolla kuuluu arvoalueeseen, ykkönen ei). Opiskelijoiden tulee miettiä, miten tätä voidaan käyttää hyväksi kolikon heitossa, jossa tulokseksi halutaan siis kruuna tai klaava. Tulos voi olla esimerkiksi:

```java
public class todennakoisyys {
    public static void main(String[] args) {
        double i = Math.random();
        if(i<0.5){
            System.out.println("kruuna");
        }else{
            System.out.println("klaava");
        }
    }

}
```

Kun opiskelijat ovat saaneet ensimmäisen ohjelman valmiiksi, lisätään siihen silmukka, jolla kolikkoa voidaan heittää niin monta kertaa kuin halutaan. Tätä voidaan pohtia taas uuden vuokaavion avulla. Vuokaavion voi tehdä sen pohjalta, miten itse suorittaisi tehtävän, jossa kolikkoa pitää heittää vaikka 20 kertaa ja lopuksi raportoida tulokset. Jos kolikkoa heitetään esimerkiksi 1000 kertaa, ei ole mielekästä, että ohjelma tulostaa kaikkien heittojen tulokset, vaan pitää miettiä, miten silmukka tallentaa jokaisen tuloksen ja tulostaa ainoastaan kruunien ja klaavojen lukumäärän.

Ohjelmaa, joka heittää kolikkoa 1000 kertaa ja tulostaa kruunien ja klaavojen määrän pyöritetään muutaman kerran ja tutkitaan tuloksia. Tulokseksi ei tule samoja määriä joka kerta, joten tässä vaiheessa on hyvä pysähtyä pohtimaan seuraavia asioita:

1. Aikaisemmin on opittu, että molempien todennäköisyys on ½, joten onko nyt vika klassisessa todennäköisyydessä, Javan Math-luokassa vai ohjelmassa? Tai ehkä ei sittenkään missään? Mitä klassinen todennäköisyys oikeastaan tarkoittaa?
2. Miten Java-luokka itseasiassa arpoo satunnaisen luvun? Mitä rajoituksia tällä on? Mitä on tässä tapauksessa satunnaisuus?

Kolikonheiton jälkeen voidaan siirtyä noppaan. Tehdään ohjelma, joka arpoo satunnaisen silmäluvun. Nyt voidaan taas miettiä, miten aikaisemmin esillä ollutta random-metodia voidaan käyttää avuksi. Lopputuloksen pitäisi näyttää kuta kuinkin tältä:

```java
int j = (int)(Math.random()*6+1);
```

Opiskelijoiden ei itse tarvitse tietää tässä vaiheessa, miten Javassa tehdään eksplisiittinen tyypinmuunnos, mutta heidän tulee itse keksiä idea.

Taas yhden onnistuneen heiton jälkeen luodaan silmukka, joka heittää noppaa monta kertaa ja kirjaa ylös tulokset. Voidaan lisäksi miettiä esimerkiksi seuraavia asioita:

1. Miten jokaiseen silmäluvun lukumäärään saataisiin myös sen todennäköisyys mukaan?
2. Nyt kun vaihtoehtoja on kahden sijasta kuusi, miten todennäköisyydet tällä kertaa eroavat klassisesta todennäköisyydestä 1/6.
3. Mitä tapahtuu, jos heittojen määrää vielä lisätään?

Seuraavaksi siirrytään kahteen nopan heittoon. Opiskelijat saavat miettiä ennen ohjelman kirjoittamista, miten laskettaisiin todennäköisyys tapahtumalle: ”Tavallista 6-sivuista noppaa heitetään kaksi kertaa. Millä todennäköisyydellä jälkimmäisellä heitolla saadaan suurempi silmäluku kuin ensimmäisellä?”. Opiskelijoiden tulee miettiä tapauksia läpi kerto- ja yhteenlaskusääntöjen avulla. Tässä kohtaa ennen laskun kirjoittamista voidaan kirjoittaa ratkaisu sanallisesti ”ensimmäisellä tulee 1 **ja** toisella 2-6 **tai** ensimmäisellä tulee 2 **ja** toisella 3-6 **tai**…”

Tämän jälkeen kirjoitetaan ohjelma, joka heittää noppaa kaksi kertaa ja sen jälkeen kirjaa ylös, oliko jälkimmäinen suurempi. Tehdään tästä silmukka ja kokeillaan 1000 kertaa, 10000 kertaa jne. Voidaan vertailla tuloksia aikaisemmin laskettuun todennäköisyyteen. Tässä kohtaa on hyvä pohtia:

1. Kuinka monta silmukkaa pitää tehdä, että tulos on lähellä aikaisemmin laskettua tulosta?

2. Kuinka monta toistoa yleensäkin riittää osoittamaan jokin todennäköisyys? 1000? 1 000 000? Vai riittääkö mikään? Mikä on ”tarpeeksi lähellä”? Onko mahdollista, että esimerkiksi 1 000 000 toistossa ei tule yhtään kuutosta?

Jos jotkut opiskelijat ovat nopeita selvittämään nämä tehtävät, voidaan haastetta lisätä ohjelmalla, joka arpoo satunnaisen pelikortin. Tässä pitää pohtia esimerkiksi sitä, kannattaako arpoa erikseen maa ja kortin arvo.

Harjoituksessa on tarkoitus pohtia, mitä satunnaisuus oikeasti tarkoittaa ja miten satunnaisuudesta puhutaan puhekielessä. Ovatko nämä sama asia? Yleensä kyselen MAA10-kurssin aluksi, minkälaisia pelejä opiskelijat itse pelaavat tai ainakin tietävät säännöt. Tässä kohtaa voidaan laajentaa pohdintaa tuttuihin peleihin: Minkälaista satunnaisuutta liittyy itselle tuttuihin peleihin? Miten esimerkiksi netissä toimivat kasinot toimivat? Onko mahdollista löytää tietoa kasinoiden pelien tavasta tuottaa satunnaisia kuvakkeita? Tämä kohta on siis varsin vapaata tutkimusta oman mielenkiinnon mukaan. Omilla kursseilla ainakin tähän mennessä jokainen on netissä pelaillut jotain peliä, mihin liittyy todennäköisyydet. Esimerkkejä ovat muun muassa peleissä ostettavat yllätyslaatikot (loot box), joista saa esineitä tai tehosteita peliin.

Jos näyttää siltä, että ryhmässä on opiskelijoita, joilta helpoimmat ohjelmointiharjoitukset sujuvat nopeasti, heille voi keksiä lisää puuhaa. Esimerkiksi:

1. Peli, jossa pelaajalta kysytään ”kruuna vai klaava?” ja tämän jälkeen ohjelma kertoo, voittiko tietokone vai pelaaja. Vaikeampana versiona silmukka, jossa peli jatkuu, kunnes pelaaja syöttää tyhjän. (Tässä voidaan myös pohtia sellaista tilannetta, että pelaaja ei seuraa pelin ohjeita ja syöttää jotain muuta, tai että pelaaja kirjoittaa vahingossa vaikkapa ”KRuuna”.)
2. Tai sitten jokin muu peli – opiskelija voi myös itse ehdottaa, mitä hän haluaisi tällä todennäköisyysharjoituksella tehdä

Tavoitteet ja arviointi: Opiskelija ymmärtää, millainen algoritmi esimerkiksi silmukassa toistuu. Tätä voidaan miettiä sen kautta, miten asian tekisi itse: ”nopan heitto – kirjaus – nopan heitto – kirjaus…” ja lopuksi tulosten raportointi ja tulkinta. Huomioidaan tässä muun muassa, että mitä tarvitaan aluksi (noppa ja välineet kirjaukseen). Sen jälkeen pohdinta, miten tämän kertoo tietokoneelle.

Opiskelija miettii satunnaisuuden käsitettä: Onko esimerkiksi nettipeleissä oikeaa satunnaisuutta? Tässä voidaan myös miettiä suojattua lähdekoodia. Jos vaikka nettikasinolla on suojattu lähdekoodi, ei ole mitään takeita siitä, että algoritmissa on edes pseudosatunnaisuutta.

Mielestäni työskentelystä ei tarvitse antaa arvosanaa. Opiskelijoita pitää myös arvioida eri tavalla jokaisen omasta lähtötasosta käsin. Mielestäni tärkeintä on työhön tarttuminen ja aito asioiden pohtiminen. Vaikka koodin kirjoitus ei ensikertalaiselta niin sujuisikaan, tärkeää on rohkeasti yrittää sekä miettiä tehtävien oheen annettuja kysymyksiä ja etsiä myös vastauksia kysymyksiin, jotta satunnaisuuden käsite ei jää mutu-tasolle. Opiskelijoille voisi työskentelyn päätteeksi tehdä itsearviointilomakkeen, jossa he saavat itse miettiä, mitä he oppivat harjoituksesta. Arviointien pohjalta voidaan vaikka vielä seuraavalla tunnilla käydä purkukeskustelu, jos opettajan mielestä palautteet näyttävät siltä, että niistä voisi vielä irrota lisää keskustelua.

Eukleideen algoritmi, lukio

Milla Hirvonen, CC BY-SA 4.0

6.2.2021

Tuntisuunnitelma – Eukleideen algoritmi

Milla Hirvonen – 2021

Aihepiirin valinta ja rajaus

Tässä tuntisuunnitelmassa esitellään noin kahden oppitunnin mittainen paketti Eukleideen algoritmin opiskeluun ja kyseisen algoritmin ohjelmoimiseen. Ohjelmointia käytetään siis matematiikan opiskeluun (erityisesti uuden LOPS:n moduulissa MAA11 Algoritmit ja lukuteoria). Eukleideen algoritmi on nimensä mukaisesti algoritmi, jota käytetään kahden luvun suurimman yhteisen tekijän löytämiseksi. Kyseinen algoritmi on mainittu keskeisenä sisältönä MAA11-moduulissa. Lisäksi moduulin yhtenä tavoitteena on, että opiskelija oppii toteuttamaan yksinkertaisia algoritmeja ohjelmoimalla. Nämä tavoitteet varmasti motivoivat opiskelijoita, esimerkiksi siksi, että jos asiat on mainittu opetussuunnitelmassa, niitä voidaan kysyä ylioppilaskokeissa. [1], [4]

Tuntisuunnitelman pohjatietoina matematiikan osalta ovat aiemmat lukion kurssit. Lisäksi oletetaan, että opiskelijat ovat jo jossain määrin perehtyneet siihen, mikä vuokaavio on. Ohjelmoinnin puolella ennakkotietona vaaditaan, että opiskelija on tutustunut tekstipohjaiseen ohjelmointiin, mutta ei välttämättä ole tehnyt sitä paljoa.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Matematiikan oppimistavoitteet:

- Jakoyhtälön kertaaminen
- Eukleideen algoritmin käytön opettelu
- Vuokaavion piirtäminen

Ohjelmoinnin oppimistavoitteet:

- Muuttujien käyttö
- While-silmukka
- Syötteen tarkastus (ylöspäin eriyttävä)

Matematiikan osalta opettaja voisi arvioida opiskelijan tuottaman vuokaavion oikeellisuuden. Ohjelmoinnissa arvioidaan sitä, tekeekö ohjelma sen mitä pitää. Plussaa arvioinnissa saa siitä, jos oma koodi on kommentoitua ja jos se on mahdollisimman tehokas. Näiden kahden oppitunnin tehtävistä siis palautetaan vuokaavio sekä itse ohjelmakoodi. Vaikka matematiikka ja ohjelmointi periaatteessa arvioidaan erikseen, ne ovat kuitenkin kiinteä osa toisiaan, ja matematiikka on tavallaan ”johdantona” ohjelmoinnille.

Opiskelija itse tekee ennen tuntien alkua itsearvioinnin omasta ohjelmointiosaamisestaan (mitä osaat ennestään, mikä on ohjelmoinnissa ollut hauskaa, mikä haastavaa, mitkä ovat tavoitteet tälle oppimiskokonaisuudelle). Kokonaisuuden lopuksi opiskelija arvioi, kuinka hyvin saavutti asettamansa tavoitteen sekä kertoo omin sanoin, mitä hän on oppinut tuntien aikana.

Työskentelyvälineet ja opetusmenetelmät

Opiskelijan ohje löytyy osoitteesta https://colab.research.google.com/drive/1WWzKkkm9kefngxBczyeJhuRUUS7lF4Qh?usp=sharing. Ohjeessa on esimerkkiratkaisu vuokaavion sekä mallikoodi. Näitä ei ole tarkoitus jättää ohjeeseen opiskelijoille. Ohjeiden lähteinä on käytetty [2], [3], [4] ja [5].

Tehtävät suositellaan tehtäväksi Googlen Colab -ympäristössä Python-kielellä. Tämä sen takia, ettei koneelle tarvitse välttämättä asentaa mitään. Lisäksi Colabissa on helppo lähteä työstämään annettua työkirjaa ja työ pysyy helposti tallessa. Kaikilla opiskelijoilla on koulun tunnukset, jotka toimivat myös Googlessa.

Ensimmäinen tunti alkaa Eukleideen algoritmin esittelyllä. Tässä yhteydessä kerrataan myös, mitä jakoyhtälö tarkoittaa. Opiskelijat tekevät käsin muutaman tehtävän, joissa etsitään kahden luvun suurin yhteinen tekijä. Apuna voi käyttää myös oppikirjaa ja sen tehtäviä. Tämän jälkeen opiskelijoiden tehtävänä on luoda vuokaavio algoritmille. Vuokaavioita voi tehdä vaikka minkälaisesta toiminnasta, ja tämä välivaihe matemaattisen algoritmin ja ohjelmoinnin välissä lisääkin mahdollisuutta siihen, että opiskelija alkaa nähdä arkielämässä toimintoja, joita voisi hoitaa sopivan ohjelmanpätkän avulla. Yhteistyön tekeminen kaikissa tämä työskentelyn vaiheissa on erittäin suotavaa, ja sitä tapahtuu varmasti luonnostaan. On kuitenkin toivottavaa, että jokainen opiskelija tekee oman tuotoksen. Varsinkin ohjelmointi on sellaista, että katsomalla toisen tekemistä ei omaan mieleen välttämättä jää kauheasti.

Toisella oppitunnilla siirrytään ohjelmoimaan. Tällä kertaa opiskelijat tutustuvat ohjeessa oleviin mallikoodeihin ja alkavat niiden avulla muokata vuokaaviotaan ohjelmakoodiksi. Tunnin lopuksi opiskelijat voisivat verrata esim. pareittain tekemiään ohjelmia. Niissä on kenties pieniä eroja, mutta ne voivat kuitenkin molemmat olla toimivia. Tästä päästään keskustelemaan vaikka koko ryhmän kanssa yhteisesti eri tapojen vahvuuksista ja heikkouksista. Jos opiskelija osaa ohjelmoida sujuvasti ja saa ohjelman perustoiminnallisuuden tehtyä nopeasti, hän voi jatkaa ohjelman kehittelyä syötteen tarkistuksella. Mitä jos annettu luku ei olekaan kokonaisluku tai positiivinen? Mitä jos käyttäjä syöttää ensin pienemmän ja sitten suuremman luvun?

Lähteet

[1] Opetushallitus, *Lukion opetussuunnitelman perusteet 2019*. Saatavissa (viitattu 5.2.2021) https://www.oph.fi/sites/default/files/documents/lukion_opetussuunnitelman_perusteet_2019.pdf

[2] Wikipedia, *Eukleideen algoritmi*. Saatavissa (viitattu 5.2.2021) https://fi.wikipedia.org/wiki/Eukleideen_algoritmi

[3] Wikipedia, *Jakoyhtälö*. Saatavissa (viitattu 5.2.2021) https://fi.wikipedia.org/wiki/Jakoyht%C3%A4l%C3%B6

[4] P. Heiskanen, P. Kaakinen, J. Lehtonen, M. Leikas, J. Tahvanainen, *Tekijä Pitkä matematiikka 11 Lukuteoria ja todistaminen*. SanomaPro. 2020.

[5] Wikipedia, *Vuokaavio*. Saatavissa (viitattu 5.2.2021) https://fi.wikipedia.org/wiki/Vuokaavio

Geometrinen todennäköisyys, lukio

Miika Karpin, CC BY-SA 4.0

8.10.2021

TUNTISUUNNITELMA LUKIOON ASIAAN GEOMETRINEN TODENNÄKÖISYYS ELI LIITTYEN KURSSIIN MAA10 (LOPS 2016) TODENNÄKÖISYYSLASKENTA JA TILASTOT

Opiskelijat ovat suorittamassa pitkän matematiikan viimeistä pakollista kurssia MAA10 Todennäköisyys ja tilastot. Keväällä 2021 pitkän matematiikan YO-kokeessa oli ensimmäistä kertaa tehtävä, jossa ratkaisun sai tekstin mukaan koodata. Käytännössä tämä tarkoitti Geogebralla ja/tai Libre Calcilla tehtävän suorittamista mutta myös TI-nspire ohjelmiston käyttö oli mahdollista YO-kokeessa. TI ohjelmisto sisältää Python kääntäjän ja Abitti-ympäristössä oleva selain toimii myös Javascriptin ajoon kun tiedoston tallentaa .html muodossa. Lukiossa tunnit ovat 75 min mittaisia eli on melkein ns. kaksoistunnin mittainen 2*45 min peruskoulussa. Kokonaisuus pitäisi pystyä vetämään yhdellä 75 min oppitunnillakin, kun taustatietoina on todennäköisyyslaskennan perusteet ja ns. klassisen todennäköisyyden määritelmä. Aihepiirinä tarkemmin ottaen ko. tunnilla on ns. geometrinen todennäköisyys. Geometrisessa todennäköisyydessä tarkoitetaan, että suotuisat alkiot voivat olla myös suotuisa pituuksia, pinta-aloja tai tilavuuksia. Suotuisalla viitataan tässä kysytyn tapahtuman A kannalta suotuisiin tapauksiin. Esim. nopan heitossa tapahtuman A=”tulee vähintään silmäluku 5” suotuisia tapauksia on 2 kpl (silmäluvut 5 ja 6), jolloin todennäköisyys lasketaan 2/6=1/3. Tässä ns. klassisessa todennäköisyydessä jaetaan suotuisat tapaukset tapahtumaan liittyvillä kaikilla alkeistapauksilla (silmäluvut 1-6 eli 6 kpl). Useimmat opiskelijat ovat käyneet myös edellisessä jaksossa kurssin MAA12 Algoritmit matematiikassa, jonka kautta muutamat silmukat ja Python alkeet ovat tuttuja. Geogebran käyttö on tuttua kaikille lukiossamme, mutta TI nspire CALC ohjelmisto on taas vähemmän tunnettu eivätkä kaikki ole sitä asentaneet omalle koneelleen.

Mutta toki tätä voisi käsitellä myös kahdella oppitunnilla, jos tunnilla on opiskelijoita, jotka eivät ole käyneet kurssia MAA12. Tällöin esiteltäisiin muutama luku tie.koodariksi.fi aineistosta, jotta Python koodin tai koodaamisen perusteiden läpikäymiseen ei mene sitten liikaa aikaa. Toisaalta myös koska aikaa menee ohjelmien tekemiseen ja testaamiseen eri opiskelijoilla eri määrä niin 2*75minuuttiakaan ei ole kaikille liian pitkä aika. Opettajan ajatuksena on käsitellä tätä alla olevaa koodaamiseen liittyvää pitkän matematiikan tehtävää Geogebralla, Python ohjelmoinnilla, TI-nspire CALC, Javascriptilla ja Libre Calcilla eli MS Exceliä vastaavalla ilmaisohjelmalla. Näiden kaikilla em. voidaan ratkaisu tuottaa ns. Abitti-ympäristössä, jota YTL käyttää ylioppilaskirjoituksissa. Joku näistä em. ohjelmista voidaan toki jättää pois jos tulee tunnilla/tunneilla liian kiire ja keskittyä enemmän niihin vaihtoehtoihin joita lukiossa käytetään.

Arvioinnista

Suorituksia voidaan arvioida kahdesta näkökulmasta eli matemaattisesta ja tietoteknisestä. A-kohdassa matematiikan puolella arvioidaan onko oikeasta alueesta valinnut suotuisat pisteet ja suhteuttaako sen koko tutkittavaan pinta-alaan. Ohjelmoinnin puolella taas arvioidaan onko opiskelija suorittanut oikeat käskyt oikeilla parametreilla ja onko tehty testiajoja ohjelman suorituksesta mahdollisen virheen löytämiseksi. B-kohdassa taas tarkastellaan matemaattisesti keskiarvon laskemista ja todennäköisyyden antamista suhteessa suorakaiteen alaan, kun taas ohjelmoinnin puolella tarkastellaan keskiarvon laskemiseksi tehtyä algoritmia. YTL on julkistanut seuraavat pisteytysohjeet tästä tehtävästä:

8.

1000 lukuparin arpominen oikeilla väleillä JA komento tai tarkka selitys	2+1
ehdon testaus JA komento tai tarkka selitys.	1+1
lukumäärän laskeminen.	1
Esimerkki taulukkolaskentaratkaisusta. Eri arvontakerroilla saadaan eri luvut. Oheinen on siis vain yksi esimerkki mielekkäistä luvuista, jotka taulukointi voi antaa.	
Esimerkkikoodi:	

```
1  import random
2  n=1
3  k=0
4  while n<1001:
5      x=random.uniform(0,2)
6      y=random.uniform(0,4)
7      if x*x<=y:
8          k=k+1
9      n=n+1
10 print(k)
```

Pseudokoodiratkaisu	max6
Ratkaisuissa on käytetty toisinaan kokonaislukujen arpomista. Arpomalla kokonaisluvut isolta väliltä ja skaalaamalla voidaan rakentaa täysien pisteiden ratkaisu. Toisaalta on myös arvottu desimaalilukuja, mutta asetukset valittu niin, että vain kokonaislukuosa näkyy.	
Taulukossa näkyy vain kokonaislukuja, mutta ratkaisusta ilmenee, että on käytetty desimaalilukuja.	max6
Taulukossa näkyy vain kokonaislukuja välillä 0–4, eikä mistään ilmene, että on käytetty desimaalilukuja.	max4
Arvottu kokonaislukuja + skaalattu.	max6
Keskiarvo 665.	1
Perusteluna komento tai lauseke.	1
▽ Ala on on siis noin $\frac{665}{1000}$ koko suorakulmiosta,	(2)
eli $\frac{665}{1000} \cdot 2 \cdot 4 \approx 5{,}3$.	2
Laskettu pinta-ala käyttämättä asianmukaisesti Hillen ohjelman tulosteita.	0

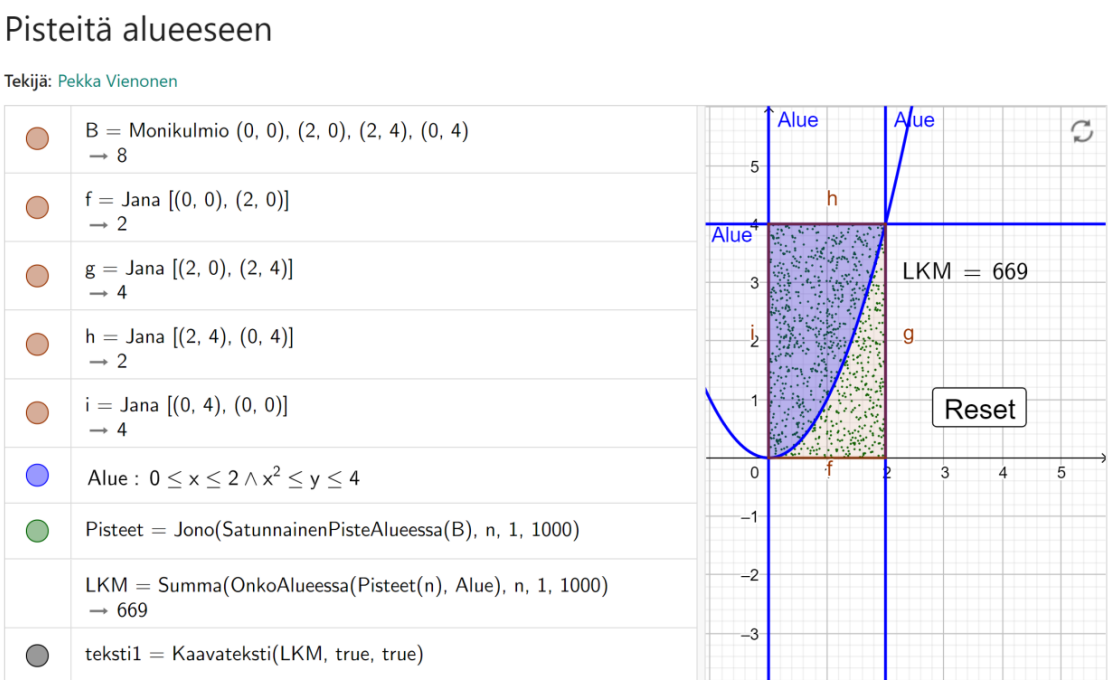
Tehtävänanto kirjaimellisesti YO K21/8:

Tasojoukon A pisteet (x,y) määräytyvät epäyhtälöistä 0<=x<=2,0<=y<=4 ja $y>=x^2$. Tässä tehtävässä tarkoitus on arvioida joukon A pinta-alaa simulaation avulla käyttämällä sitä tietoa, että todennäköisyys on suoraan verrannollinen pinta-alaan. Arvotaan pisteitä (x,y) suorakulmiosta B, jonka määräävät epäyhtälöt 0<=x<=2 ja 0<=y<=4.

1. Tee sopivalla ohjelmistolla koodi, joka arpoo 1000 pistettä suorakulmiosta B ja tulostaa vastauksena niiden pisteiden lukumäärän, jotka kuuluvat joukkoon A. Kerro sanallisesti ja sopivien kuvakaappausten avulla, miten toteutit koodisi. (Vihje: voit käyttää esimerkiksi taulukkolaskennan satunnaislukugeneraattoria.) (6 pistettä)
2. Hille ajoi kohdassa 1 tekemänsä koodin 10 kertaa ja sai alla olevat luvut. Laske tulosten keskiarvo ja arvioi tämän perusteella joukon A pinta-alaa (6 pistettä). Hillen koodin tulosteet: 673,664,672,679,667,650,640,678,660,667.

Tapa 1. Geogebra

Kirjoita Algebra-ikkunaan seuraavat käskyt tai käytä Geogebran valikoita esim. monikulmion ja janan muodostamiseen. Suomen kielen ja-sanaa vastaa Geogebrassa merkintä && ja sitä vastaava looginen konnektiivi on tuo ylösalaisin V. Luodaan 1000 kpl pisteitä jono-komennolla, jossa viitataan alueeseen B. Alue B on aluksi luotu suorakaidealue, jota rajoittavat pisteet (0,0),(2,0)(2,4) ja (0,4). Seuraavaksi LKM muuttujaan tallennetaan yhteenlasketut ehdot toteuttavat pisteet ja näytetään tulos myös koordinaatistossa.

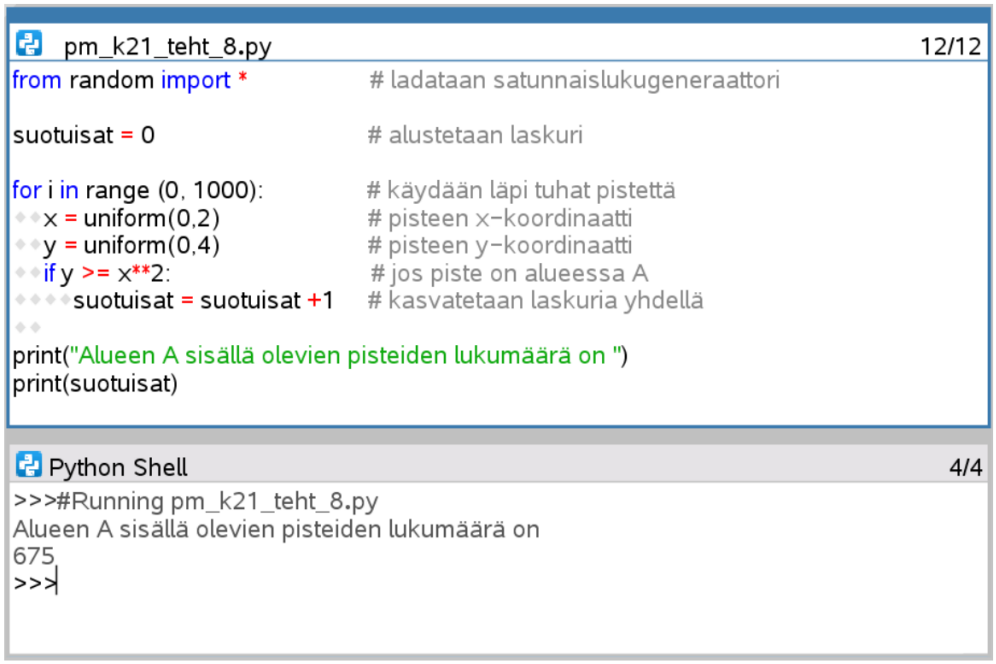


Tapa 2. Python

Peruseditori (esim. tie.koodariksi.fi/editori)

```
import random

n=1

k=0

while n<1001:

        x=random.uniform(0,2):

       y=random.uniform(0,4)

       if (x*x <= y):

           k=k+1

       n=n+1

print(k)
```

tai ajetaan koodia jollakin Python-kääntäjällä:



Tapa 3. Libre Calc

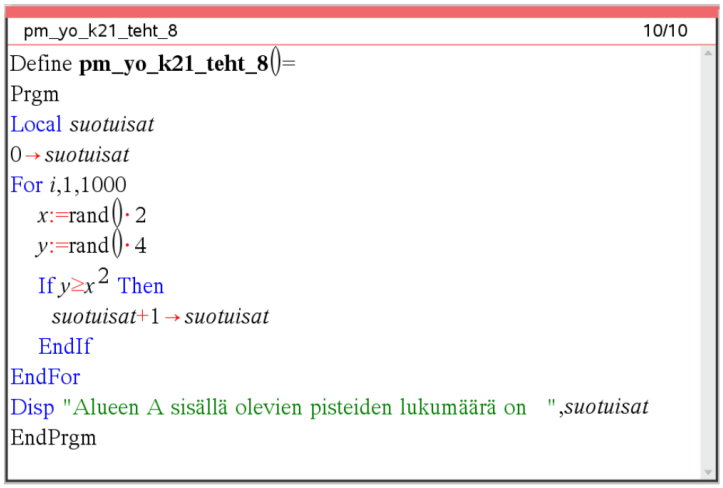
Arvotaan sarakkeisiin A ja B käskyllä SATUNNAISLUKU()*2 luku väliltä 0-2 ja käskyllä SATUNNAISLUKU()*4 luku väliltä 0-4 kuten pyydettiin. Testataan sarakkeessa C onko arvottu piste halutulla alueella JOS-käskyllä. Jos piste on alueessa niin solu saa arvon 1 ja jos ei ole niin solu saa arvon 0. Sarakkeeseen D lasketaan montako suotuisaa tapausta saatiin summaamalla sarakkeen C arvot yhteen.

	A	B	C	D
1	**1000 kpl suorakulmion B sisällä olevia pisteitä**			
2	***x***	***y***	**Onko joukon A piste**	**Yhteensä joukon A pisteitä**
3	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B3>=A3^2;1;0)	**=SUMMA(C3:C1002)**
4	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B4>=A4^2;1;0)	
5	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B5>=A5^2;1;0)	
6	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B6>=A6^2;1;0)	
7	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B7>=A7^2;1;0)	
8	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B8>=A8^2;1;0)	
9	=SATUNNAISLUKU()*2	=SATUNNAISLUKU()*4	=JOS(B9>=A9^2;1;0)	

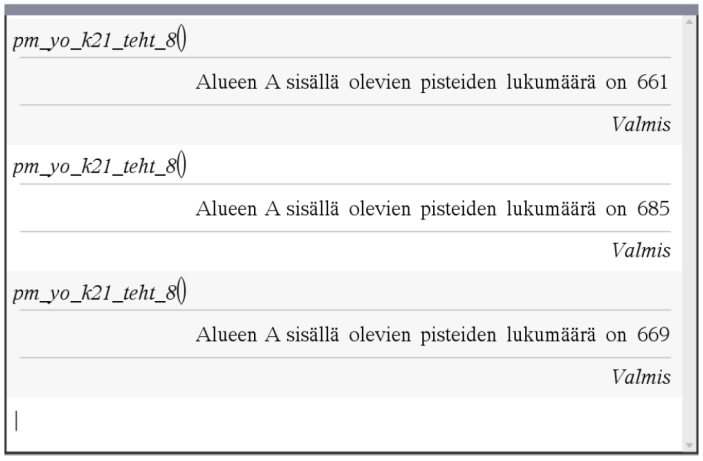
	A	B	C	D
1	**1000 kpl suorakulmion B sisällä olevia pisteitä**			
2	***x***	***y***	**Onko joukon A piste**	**Yhteensä joukon A pisteitä**
3	0,831148930182113	0,42890108430439	0	**660**
4	1,65364308949706	0,942192110754995	0	
5	0,3926416840672	3,98797178998551	1	
6	0,871126274281129	2,91010143789164	1	
7	0,946970068411052	2,2874628800116	1	
8	1,56536403563606	1,02281020514969	0	
9	1,67374432570116	3,77390160422171	1	

Tapa 4. TI-nspire CAS

Ohje avaa muistiinpanot ja kirjoita seuraava koodi. Tallenna esim. nimellä pm_yo_k21_teht_8.



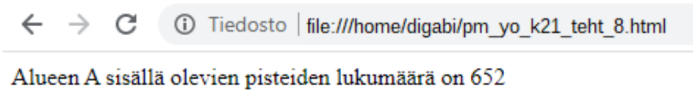
Ajetaan ohjelma muutaman kerran:



Tapa 5. Javascript

Ohje: Avaa Mousepad-ohjelma (tai muu vastaava yksinkertainen tekstieditori) ja tallenna nimellä alla oleva koodi siten, että tiedostopääte on .html. Avaa selaimella tämä html-tiedosto.

```
<script>

var suotuisat = 0;              // alustetaan muuttuja
var x = 0, y = 0;               // alustetaan muuttujat

for (i = 0; i < 1000; i++){     // for-silmukka, 1000 toistoa
  x = Math.random() * 2;        // pisteen x-koordinaatti
  y = Math.random() * 4;        // pisteen y-koordinaatti
  if ( y >= x**2 ) {            // jos piste on alueessa A
     suotuisat++;               // kasvatetaan laskuria yhdellä
  }
}

document.write(`Alueen A sisällä olevien pisteiden lukumäärä on ${suotuisat}`);
      // tuloksen kirjoittaminen
</script>

```



Hillen saaman tuloslistan keskiarvon voit perinteisen keskiarvon kaavan sijaan laskea myös esim. Libre Calcilla käskyllä =KESKIARVO(673;664;672;679;667;650;640;678;660;667) => 665.

Lähteet:

Mafynetti.fi pitkä matematiikka K2021 ratkaisut

Yle Abitreenit pitkä matematiikka K2021 Lopulliset hyvän vastauksen piirteet.

Pekka Vienonen Geogebra-appletti (https://www.geogebra.org/m/czfvx8pg)

Tehnyt/koostanut: FM Miika Karpin, Sotungin lukio/etälukio

Pythonin alkeiden harjoittelua, lukio

Pirjo-Riitta Elo, CC BY-SA 4.0

29.9.2022

LUMATIKKA 3

Algoritmisen ajattelun kehittäminen

OHJELMOINNIN OPETUKSEN SUUNNITTELU

Aihepiirin valinta ja rajaus

Oma ohjelmointitaustani on vähäinen ja lukion uudessa opetussuunnitelmassa (LOPS2021) ohjelmointi on osa valtakunnallista MAA11-opintojaksoa. Lukiossamme kyseinen opintojakso on tarjolla ensimmäisen kerran huhti-toukokuussa 2023, joten ennalta suunniteltuja oppitunteja opintojaksolle minulla ei vielä ole valmiina. Siksi ajattelin, että nyt on hyvä tilaisuus aloittaa suunnittelu, tutustua uuteen materiaaliin ja opiskella Python-ohjelmointia.

Aloitan suunnittelun ohjelmointijakson alkupuolelta, jossa tavoitteena on yksinkertaisen algoritmin toteuttaminen Python-kielellä.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppimistavoitteet:

- Opiskelija pystyy tekemään vähintään Tie koodariksi-sivuston 1. osan tehtävät (MAA11).
- Opiskelija osaa käyttää print- ja input komentoa.
- Opiskelija ymmärtää, mitä ohjelmoinnissa tarkoittaa muuttuja, miten se määritellään ohjelmassa ja miten se eroaa matematiikassa käytetystä muuttujasta.
- Opiskelija ymmärtää, millaisia lukuja Python-kielessä lyhenteet int ja float tarkoittavat.
- Opiskelija osaa käyttää Python –kielen tavallisimpia laskuoperaatioita.

Arviointi:

- Koska kyse on yksittäisestä oppitunnista ja ohjelmointiin tutustumisesta, oppitunnin hallinta arvioidaan numeerisesti kokonaisuutena vasta opintojakson lopussa pidettävällä kokeella.
- Työskentelyssä pyritään parityöskentelyyn, jolloin opettajalla riittää paremmin aika kiertää luokassa ja antaa henkilökohtaista suullista palautetta opiskelijoille. Palaute pyrkii olemaan kannustavaa sekä tarvittaessa tehtävää eteenpäin vievää.

- Käyttämässämme oppikirjassa on kunkin luvun viimeinen tehtävä suunniteltu itsearvioinnin tueksi. Eli opiskelija tekee tehtävän, katsoo malliratkaisun ja pisteyttää oman ratkaisunsa.
- Reflektointia tulee parityöskentelyssä luontaisesti.

Työskentelyvälineet ja opetusmenetelmät

Oppikokonaisuudessa käytetään välineinä tietokoneita, SanomaPron oppikirjaa MOODI 11: Algoritmit ja lukuteoria, Tie koodariksi –sivustoa sekä kynää ja paperia.

Uskon oppilaiden motivoituvan aiheesta jo ihan siksi, että ohjelmointi on erilaista matematiikkaa ja siksi, että useat harrastavat pelaamista.

Oppitunnilla on tarkoitus työskennellä sekä omatahtisesti että pareittain. Opettaja kiertää luokassa opastamassa opiskelijoita tarpeen mukaan.

Ne opiskelijat, joilla on ohjelmointitaustaa ennestään, voivat tehtävien ratkomisen sijaan toimia apuopettajina tai siirtyä suoraan oppikirjan haastaviin tehtäviin.

Kunhan ohjelmoinnissa päästään kunnolla vauhtiin, voidaan oppitunneille tuoda esimerkkejä muista oppiaineista ja muilta matematiikan opintojaksoilta (esim. Todennäköisyyslaskenta) ja näin kannustaa opiskelijoita hyödyntämään taitojaan myös MAA11-oppituntien ulkopuolella.

Tunnin kulku

Aloitetaan ohjelmointiin tutustuminen Tie koodariksi -sivuston MAA11-opintojaksolle tarkoitetuilla tehtävillä. Tehtäviä voidaan antaa myös kotitehtäväksi edellisellä tunnilla aiheeseen orientoitumiseen. Ennen oppituntia luon sivustolle ryhmän MAA11 / Pirjo-Riitta, johon opiskelijat liittyvät. Näin voin seurata heidän etenemistään ja auttaa apua tarvitsevia paremmin.

Koska oppitunnilla edetään omatahtisesti, seikkaperäiset ohjeet on hyvä jakaa opiskelijoille esim. Teamsin välityksellä ja/tai heijastaa taululle.

Jos ryhmässä on opiskelijoita, jotka ovat aiemmin ohjelmoineet, voivat he tehtävien tekemisen sijaan toimia apuopettajina TAI siirtyä suoraan tekemään tehtäväsarjan II tehtäviä 9.11-9.16. Apuopettajina toimimisesta voidaan sopia jokin pistehyvitys, sillä koulussamme on ollut käytäntönä tietty pistehyvitys myös tietystä tehtävämäärästä.

1) JOHDANTO PYTHONIIN, osa 1:

Tehdään tehtävät 1-6

Harjoitellaan PRINT –komentoa, muuttujan määrittämistä sekä Pythonissa käytettäviä laskuoperaattoreita (+ , - , / , // , * , ** ja %)

Tehtäviä voidaan tehdä omatahtisesti, sillä ohjeistus on hyvin selkeä ja ohjelmaeditori on integroitu sivustolle. Myös automaattitarkistus on mukana.

2) TEORIAA:

Jos Pythonissa määritellään

Luku1 = int()

Luku2 = float()

Luku3 = str()

Miten ohjelma lukee/tulkitsee kunkin luvun?

Millä käskyllä desimaaliluvun pyöristäminen taphtuu?

VASTAUS:

Luku1 on kokonaisluku, Luku2 desimaaliluku ja Luku3 merkkijono.

Pyöristäminen tapahtuu käskyllä **round()**.

3) HARJOITUS: (Oppikirjan tehtävää 9.1 muokattuna)

Suunnittele ohjelma, joka kysyy käyttäjältä kaksi kokonaislukua ja tulostaa niiden summan.

Annetaan opiskelijoille osittainen ohjelmakoodi alku, jonka täydentävät loppuun testaten sen toimivuutta itse.

```
#Kysytään luku1

luku1 =

#Kysytään luku2.

luku2 =

#lasketaan lukujen summa

summa =

# Tulostetaan summa.

print
```

OHJEET:

Funktio **input()** lukee syötetyn luvun merkkijonona, eli ei lukuna. Tutki ensin, millaisen lopputuloksen tällä käskyllä saat.

Funktio **int()** muuttaa merkkijonon kokonaisluvuksi.

Kokeile, mikä ero on lopputuloksessa, jos kirjoitatkin **int(input()).**

RATKAISU:

```
#Kysytään luku1

luku1 = int(input("Anna joku kokonaisluku."))

#Kysytään luku2.

luku2 = int(input("Anna toinen kokonaisluku"))

#lasketaan lukujen summa

summa = luku1 + luku2

# Tulostetaan summa.

print(summa)
```

4) HARJOITUS: (Oppikirjan tehtävää 9.8 muokattuna)

Suunnittele ohjelma, joka kysyy käyttäjältä kolme kokonaislukua ja tulostaa niiden keskiarvon.

RATKAISU:

```
#Kysytään luvut

luku1 = int(input("Anna joku kokonaisluku."))

luku2 = int(input("Anna toinen kokonaisluku"))

luku3 = int(input("Anna kolmas kokonaisluku"))

#lasketaan lukujen keskiarvo

keskiarvo = (luku1 + luku2 + luku3)/3

# Tulostus

print(keskiarvo)
```

5) Oppikirjan tehtävä 9.4

Tee ohjelma, joka kysyy käyttäjältä ympyrän säteen, lukee sen, laskee ympyrän piirin ja pinta-alan ja tulostaa ne näytölle pyöristettynä kahden desimaalin tarkkuuteen. Käytä piille likiarvoa 3,14.

RATKAISU:

```
#Kysytään säde

säde = float(input("Anna ympyrän säde."))

#Lasketaan ympyrän piiri ja pinta-ala.

piiri = round(2*3.14*säde,2)

ala = round(2*3.14*säde**2,2)

# Tulostus

print("Ympyrän piiri on",piiri)

print("Ympyrän pinta-ala on",ala)
```

6) Tehtävä 9.5

Tee ohjelma, joka tulostaa osamäärän ja jakojäännöksen, kun käyttäjän syöttämä positiivinen kokonaisluku jaetaan luvulla 7.

LISÄKYSYMYS: Tutki, mitä tapahtuu, kun käyttäjä syöttää negatiivisen kokonaisluvun.

RATKAISU:

```
#Kysytään luku
```

```
luku = int(input("Anna positiivinen kokonaisluku."))
#Lasketaan luvun kokonaisosa ja jakojäännös
kokonaisosa = luku // 7
jakojäännös = luku % 7
# Tulostus
print("Kokonaisosa on",kokonaisosa)
print("Jakojäännös on",jakojäännös)
```

Jos syötetään esim. Luku 20, saadaan osamäärä 2 ja jakojäännös 6, sillä

20 = 7*2 + 6

Jos syötetään esim. luku –10, saadaan tulokseksi osamäärä -2 ja jakojäännös 4, sillä

-10 = 7*(-2) + 4

KOTONA:

TEHTÄVÄ 9.3 ja 9.10 (perustehtäviä)

TAI 9.6 ja 9.7 (vaativampia tehtäviä)

SEKÄ

KAIKKI tekevät Taitopuntari 9:n

Python-ohjelmointia, lukio

Maiju Mäenpää, CC BY-SA 4.0

28.9.2022

Aihepiirin valinta ja rajaus

- **Oppiaine:** Matematiikka
- **Aihepiiri**

  Algoritminen ajattelu, algoritmien tulkinta, Python-ohjelmointi ja vuokaaviot. Harjoitus on suunniteltu MAA11-kurssille.

  Harjoitus voidaan toteuttaa jo aiemmassa vaiheessa muillakin kursseilla, esimerkiksi funktioiden tulkintaa harjoitellessa ja sitä on mahdollista hyödyntää myös lyhyessä matematiikassa, kun funktioista on käyty 2. asteen polynomifunktion ratkaisua ja eksponenttifunktioita. Harjoituksessa onnistuminen ei edellytä aiempaa koodiosaamista, vaan koodia tulkitsemalla ja ajamalla oppii tunnistamaan koodin perustoiminnot.

- **Keskeiset ilmiöt**

  Algoritmeja voidaan hyödyntää monissa erilaisissa arkipäivän sovelluksissa. Tässä tuntisuunnitelmassa yhdistetään algoritmien tulkinta, lukeminen ja koodin parantaminen yhden koodin tarkastelun ympärille.

- **Miksi**

  Algoritmien tulkintaan ja lukemiseen oppii vähitellen. Lisäksi jos on ohjelmoinut vähemmän, voi tässä harjoituksessa valmista koodia tulkitsemalla oppia myös uusia ohjelmointirakenteita. Koodi ei itsessään ole kovin monimutkainen, vaan siinä hyödynnetään pääosin funktioita, muuttujia ja if-else-rakenteita. Tässä harjoituksessa opiskelijat pääsevät yhdistämään matematiikassa aiemmin oppimaansa sekä vahvistamaan monilukutaitoaan.

- **Oppimistavoitteet**
  - Opiskelija tutustuu valmiiseen koodiin ohjelmointikielisenä tekstinä.
  - Opiskelija yhdistää kirjoitetun koodin merkityksiä ajettuun koodiin.
  - Opiskelija kertaa ja vahvistaa osaamistaan funktioista.
  - Opiskelija harjoittelee valmiin koodin parantamista.

Valittuun aihepiiriin liittyvien oppimistavoitteiden määrittely ja arviointi

Oppimiskokonaisuutta analysoidaan monipuolisesti oppimiskokonaisuuteen pohjautuvasti eli matematiikan, ohjelmoinnin ja tehtäväkokonaisuuden näkökulmista / osa-alueittain. Arviointi toteutetaan osittain erillisenä prosessina. Opiskelijan itsearvioinnin ja opettajan havainnoinnin pohjana käytetään seuraavalta sivulta löytyvää tavoitetaulukkoa, mistä opiskelija voi helposti ymmärtää tehtävälle asetetut tavoitteet ja johon opiskelija voi aluksi merkitä omat tavoitteensa ja tehtävän lopuksi tarkastella onnistumistaan niissä.

Tehtävän tavoitetaulukko opiskelijalle ja opettajalle

	Alle tavoitetason	**Tavoitetaso**	**Erinomainen suoritus**
Koodin lukeminen	Kirjatuista vastauksista (suullinen tai kirjallinen) ei käy ilmi opiskelijan tekemät tulkinnat. Tehdyt tulkinnat eivät ole yhdistettävissä koodiin.	Koodista on löydetty keskeiset osat ja toiminnalliset virheet. Opiskelija on havainnollistanut koodin toimintaa vuokaaviolla.	Koodista on löydetty keskeiset osat ja virheet (toiminnalliset ja tietyissä tilanteissa ilmenevät syntaksivirheet). Opiskelija on tulkinnut saamiaan virheilmoituksia joko aiempien tietojen, päättelyn tai Internetin avulla.
Koodin kirjoittaminen	Koodiin ei ole tehty muutoksia.	Koodista löytyneet matemaattiset virheet on korjattua. Koodiin on voinut jäädä pieniä virheitä/bugeja, joiden seurauksena ohjelma kaatuu väärillä syötteillä.	Kirjoitettu koodi toimii, koodiin on tehty omia lisäyksiä ja kaikki virheet on korjattu. Uusi koodi toimii aiempaa koodia paremmin.
Matemaattinen ymmärrys	Koodin matemaattisiin osiin ei ole tehty muutoksia.	Koodista on löydetty matemaattiset virheet.	Koodiin on lisätty funktion analysointiin liittyviä ominaisuuksia. Esimerkiksi haettu suoran yhtälön nollakohdat.
Tehtävä-kokonaisuus	Tehtävän ratkaisu on täysin identtinen kopio toisen opiskelijan ratkaisusta. Tehdyt tehtävät ja pohdinnat ovat epäjohdonmukaisia ja pohdinta vähäistä.	Tehdyt muutokset ja lisäykset ovat perusteltuja tehtävän aihepiiriin sopivia.	Tehtävän ratkaisuissa näkyy opiskelijan ajattelu ja luovat ratkaisut. Ohjelmakoodiin tehdyt muutokset parantavat ohjelmaa ja ne on dokumentoitu selkeästi joko koodiin tai tehtäväpohjaan.
Työskentely-taidot	Opiskelijan työskentely tehtävän eteen vähäistä. Ongelmatilanteissa opiskelija luovuttaa ja alkaa tehdä muita asioita.	Opiskelija on tehnyt tehtävän parhaan osaamisensa mukaan. Ongelmatilanteissa opiskelija keskustelee ja pohtii ratkaisua yhdessä muiden opiskelijoiden kanssa.	Opiskelijan työskentely on itsenäistä ja huolellista. Ongelmatilanteissa opiskelija pyrkii auttamaan kanssaopiskelijoita ja kannustaa heitä.

Työskentelyvälineet ja opetusmenetelmät

- **Välineet**: Tietokone, Internet, tehtäväpohjat ja luonnosteluun kynä+paperi.
- **Ohjeistus**: Tehtävämoniste + valmis koodipohja

  Ratkaisu virhetehtäviin: Toisen asteen polynomifunktion kahta nollakohtaa ratkaistaessa tulee samat juuret. Eksponenttifunktion kasvavuudessa ja vähenevyydessä on väärät rajat.

- **Motivaatio**

  Valmiin koodin muokkaamisessa opiskelijalla on enemmän aikaa koodin lukemiseen eikä tarvitse heti alkaa ohjelmoida. Tehtävässä ei edellytetä uutta matemaattista osaamista tai välttämättä edes aiempaa ohjelmointiosaamista. Koodiin tehtävät muutokset voi päätellä aiemmin tehdyn koodin pohjalta. Hyvillä testeillä opiskelija voi löytää matemaattiset virheet pelkkien havaintojensa pohjalta eli testaamalla eli koodin lukemisesta aloittaminen ei ole välttämätöntä. Lukemisen ja tulkinnan tueksi koodissa on käytännössä lähes jokainen rivi kommentoitu.

- **Yhteistyö**

  Jokainen opiskelija tuottaa oman ratkaisunsa, mutta ratkaisuista on luvallista ja suositeltavaa keskustella toisten opiskelijoiden kanssa, niin kauan kuin ratkaisut eivät ole identtisesti toistensa kopioita. Opiskelijoita tulee kannustaa auttamaan toisiaan mahdollisissa ongelmatilanteissa.

- **Aiemman osaamisen hyödyntäminen**

  Opiskelijat voivat toteuttaa luovan vaiheen tehtävät täysin omavalintaisesti. Jos opiskelijalla on aiempaa ohjelmointiosaamista, häntä voi kannustaa tekemään koodiin lisätoiminnallisuuksia esim. toiminnon ääriarvon toteuttamiselle, ohjelman muuttamiseen oliopohjaiseksi tai estämään virheellisten syötteiden aiheuttamat ohjelman kaatumiset try-except-rakenteella.

- **Opitun asian yhdistäminen muihin tilanteisiin**

  Oppimiskokonaisuuden tarkoituksena on tarjota helposti lähestyttävää ja myönteistä ohjelmointikokemusta kaikille opiskelijoille. Kokeilun jälkeen opiskelijaa voi kannustaa palaamaan tähän koodiin, koska siitä saa hyviä esimerkkejä mm. käyttäjän syötteen pyytämiseen, kuvien piirtämiseen, laskutoimitusten tekemiseen ja funktioiden luomiseen.

Tehtäväohje

Tässä tehtävässä tarkastellaan valmista koodipohjaa trinket.io -sivustolla. Koodipohjassa generoidaan funktioita, piirretään ja analysoidaan funktiota. Muokkaa ja analysoi koodia seuraavien ohjeiden mukaisesti. Voit keskustella tehtävästä yhdessä muiden opiskelijoiden kanssa, kunhan jokainen palauttaa ja tekee lopullisen työn itsenäisesti.

Lue ohjeet pikaisesti läpi ennen tehtävän aloittamista. Voit tehdä analysointi- ja ohjelmointitehtäviä haluamassasi järjestyksessä.

Aloitusohjeet

1. Valmis koodipohja on oheisessa tiedostossa.
2. Mene osoitteeseen: https://trinket.io/embed/python3/a5bd54189b
3. Poista sivustolta vanhat merkinnät. Tämän jälkeen kopioi ja liitä koodipohja.

Analysointitehtävät

4. Tutki koodin toimintaa ja laadi koodin pohjalta vuokaavio.
5. Mitä koodirakenteita tehtävässä käytetään? Kuinka monta?
6. Millaisia matemaattisia virheitä löydät koodista?
7. Saatko ohjelman kaatumaan? Millaisissa tilanteissa? Miksi ohjelma kaatuu?

Ohjelmointitehtävät (kirjaa tehtävän yhteyteen lyhyesti tekemäsi muutokset)

8. Korjaa löytämäsi virheet. (Korjaa ensisijaisesti matemaattiset virheet.)
9. Lisää nollakohtien tarkastelu 1. asteen polynomifunktiolle.

Ohjelmoinnin lisätehtävät (kirjaa tehtävän yhteyteen lyhyesti tekemäsi muutokset)

10. Paranna ohjelman toimintaa.
11. Luo koodiin lisätoiminnallisuuksia.

(Valmis koodipohja on seuraavilla sivuilla)

```
import random                                   # kirjasto satunnaislukuja ja satunnaistuksia varten
import matplotlib.pyplot as plt                 # kirjasto kuvien piirtämiseen
import numpy as np                              # kirjasto tilastomuuttujille ja matematiikka -komennoille
import time                                     # kirjasto ajan käsittelyyn ja komentojen viivästykselle


def MuodostaFunktio():
    """
    Funktio:    MuodostaFunktio()
    Kuvaus:     Generoi 1. tai 2. asteen polynomifunktion tai eksponenttifunktion arpomalla.
                Palauttaa funktion yhtälön ja määritetyt muuttujat.
    Parametrit: Ei mitään / None
    Paluuarvo:  lista-muuttuja [funktion laji, funktio, muuttujat]
    """

    funktio = random.choice(["exp", "1", "2"])  # arvotaan funktion tyyppi

    if funktio == "1":                          # Jos funktio on 1. asteen polynomi
        k = random.randrange(-5, 5)             # arvotaan kulmakerroin, väli [-5,5]
        b = random.randrange(-10, 11)           # arvotaan vakiotermi, väli [-10, 11]
        return [funktio, f"f(x) = {k}x + {b}", k, b] # paluuarvo: [funktiotyyppi, funktio, kulmakerroin, vakiotermi]

    if funktio == "2":                          # Jos funktio on 2. asteen polynomi
        a = 0                                   # asetetaan muuttujalle a alkuarvo
        while a == 0:                           # arvotaan a, b ja c, kunnes a ei ole nolla
            a = random.randrange(-2,3)          # arvotaan a, väli [-2,2]
            b = random.randrange(-10, 11)       # arvotaan b, väli [-10, 10]
            c = random.randrange(-10, 11)       # arvotaan c, väli [-10, 10]
        return [funktio, f"f(x) = {a}x² + {b}x + {c}", a, b, c] # paluuarvo: [funktiotyyppi, funktio, a, b, c]

    else:                                       # Jos funktio on eksponenttifunktio
        q = 0                                   # asetetaan kasvukertoimelle q alkuarvo
        while q==0 or q==1:                     # arvotaan q uudelleen, kunnes q ei ole nolla tai ykkönen
            q = float("{:.2f}".format(random.random()*4))  # arvotaan q 2desimaalisena liukulukuna, välille [0, 4]
        return [funktio, f"f(x) = {q}*", q]     # paluuarvo: [funktiotyyppi, funktio, kasvukerroin]


def AnalysoiFunktio(funktio):
    """
    Funktio:    AnalysoiFunktio()
    Kuvaus:     Tarkastelee funktion kulkua sekä mahdollisia nollakohtia.
    Parametrit: Lista, jonka ensimmäinen listamuuttuja on funktiotyyppi, toinen funktio
                ja loput listamuuttujat ovat funktioon liittyviä vakioita.
    Paluuarvo:  Ei mitään / None
    """
    if funktio[0] == "1":                                                   # 1. asteen polynomifunktio
        if funktio[2] > 0:                                                  # kulmakerroin > 0
            print(f"Funktio on kasvava ja sen vakiotermi on {funktio[3]}.")
        elif funktio[2] < 0:                                                # kulmakerroin < 0
            print(f"Funktio on kasvava ja sen vakiotermi on {funktio[3]}.")
        else:                                                               # kulmakerroin = 0
            print(f"Funktiolla ei ole kulmakerrointa ja sen yhtälö sievenee muotoon f(x)={funktio[3]}.")

    elif funktio[0] == "2":                                     # 2. asteen polynomifunktio, f(x) = ax^2 + bx + c
        if funktio[2] > 0:                                      # a > 0
            print(f"Funktio on ylöspäin aukeava paraabeli.")
        else:                                                   # a < 0
            print(f"Funktio on alaspäin aukeava paraabeli.")

        diskriminantti = funktio[3]**2-4*funktio[2]*funktio[4]  # lasketaan diskriminantti
        if diskriminantti < 0:                                  # D < 0, ei nollakohtia
            print("Funktiolla ei ole nollakohtia.")
        elif diskriminantti == 0:                               # D = 0, yksi nollakohta
            nollakohta = (-funktio[3])/(2*funktio[2])
            print(f"Funktiolla on yksi nollakohta: x = {nollakohta}.")
        else:                                                   # D > 0, kaksi nollakohtaa
            nollakohta1 = (-funktio[3]+np.sqrt(diskriminantti))/(2*funktio[2]) # ensimmäinen nollakohta
            nollakohta2 = (-funktio[3]+np.sqrt(diskriminantti))/(2*funktio[2]) # toinen nollakohta

            # tulostetaan nollakohdat, annetaan nollakohdat kahteen desimaaliin pyöristettynä
            print(f"Funktiolla on kaksi nollakohtaa: x = {nollakohta1:.2f} ja x = {nollakohta2:.2f}.")

    else:                                       # Eksponenttifunktio, f(x)=q^x
        if funktio[2] > 0:                      # q > 0, kasvava
            print(f"Funktio on kasvava.")
        else:                                   # q < 0, vähenevä
            print(f"Funktio on vähenevä.")
```

```
def PiirraFunktio(funktio):
    """
    Funktio:    PiirraFunktio()
    Kuvaus:     Piirtää funktion kuvaajan.
    Parametrit: Lista, jonka ensimmäinen listamuuttuja on funktiotyyppi, toinen funktio
                ja loput listamuuttujat ovat funktioon liittyviä vakioita.
    Paluuarvo:  Ei mitään / None
    """

    x = np.linspace(-5,5,100)               # luodaan vektori, jossa tasaisin välein 100 pistettä, välillä [-5,5]

    if funktio[0] == "1":                   # Jos funktio on suora
        y = funktio[2]*x+funktio[3]         # ratkaistaan y:n arvot, sijoitetaan x:n arvot funktioon f(x)=kx+b
    elif funktio[0] == "2":
        y = funktio[2]*x**2+funktio[3]*x+funktio[4] # y:n arvot, sijoitetaan x:n arvot funktioon f(x)=ax^2+bx+c
    else:
        x = np.linspace(-2,2,100)   # luodaan vektori x, jossa tasaisin välein 100 pistettä, välillä [-2,2]
        y = funktio[2]**x

    # piirretään funktio
    plt.plot(x, y, 'b', label=funktio[1])   # piirretään (x,y) sinisellä (b) ja nimetään funktio
    plt.legend()                            # lisätään funktion nimikyltti näkyviin
    plt.show()                              # asetetaan kuva näkyviin
    time.sleep(3)                           # viivästytetään koodia 3s
    plt.close()



# Tästä osiosta kutsutaan funktioita
while True:
    # pyydetään käyttäjältä syötettä, valinta kuitataan enterillä.
    syote = int(input("Syötä numero (1. arvo funktio, 2. piirrä funktio, 3. analysoi funktiota, 4: lopeta ohjelma): "))
    if syote == 1:                          # arvotaan funktio
        funktio=MuodostaFunktio()
        print(f"Funktio on {funktio[1]}.")  # tulosta muistiin tallennettu funktio
    elif syote == 2:                        # piirrä funktion kuvaaja
        PiirraFunktio(funktio)
    elif syote == 3:                        # analysoi funktion toimintaa
        AnalysoiFunktio(funktio)
    elif syote == 4:                        # lopeta käyttäjän syötteiden kerääminen
        break
    else:                                   # tuntematon syöte
        print("komentoa ei tunnistettu")

print("ohjelma päättyy")
```

Let's play Countdown!, toisen asteen ammatilliset opinnot

Erik Eriksson ja Suvi Kääriälä, CC BY-SA 4.0

14.4.2022

Erik Eriksson ja Suvi Kääriälä

Let’s play Countdown!

Harjoitus on suunnattu toisen asteen ammatillisille opiskelijoille, joiden matemaattiset ja logiikan taidot kaipaavat suuresti harjoitusta. Tämän pelin tavoite on tuoda leikkimielisellä kisailulla harjoitusta peruslaskutoimituksia, laskujärjestystä ja ryhmätyötaitoja.

Tässä harjoituksessa katsotaan millaisia loogisia operaatioita eli laskutoimituksia voidaan suorittaa, jotta saadaan haluttu lopputulos.

Idea on oikeastaan televisailu Countdownista tuttu:

Opettaja arpoo tietokoneen avustuksella (vaikka excelillä) satunnaisen kokonaisluvun väliltä 100-999, joka on tavoite. Tavoitteen lisäksi arvotaan kuusi kokonaislukua, jotka voivat olla joko “isoja” tai “pieniä”. Isot luvut voivat saada arvon 25, 50, 75 tai 100. Pienet luvut ovat väliltä 1-10. Sama luku voi tulla valituksi korkeintaan kahdesti, tämä rajoitus koskee sekä “pieniä”, että “isoja” lukuja.

Pelin pohjan voi tehdä paperilla, tai opettaja heittää päästään lukuja (ei todennäköisesti kovin satunnaista) tai ohjelmoida vaikka exceliin hienon makroja sisältävän Countdown-simulaattorin.

Kun tavoite ja muut luvut ovat selvillä, opiskelijoilla on pieninä joukkueina etukäteen sovittu aika (3 min) kokeilla summauksella, erotuksella, kertolaskuilla ja jakolaskuilla, kuinka he pääsisivät haluttuun lopputulokseen. Jokaista lukua saa käyttää laskutoimituksessa vain kerran, mutta kaikkia lukuja ei ole pakko käyttää.

Harjoituksen tai pelin kulku menee seuraavasti,
Opiskelijat saavat päättää montako “isoa” ja montako “pientä” lukua he haluavat.

> *Esim. Opiskelijat haluavat kaksi “isoa” ja neljä “pientä” lukua.*
> *Satunnaislukugeneraattori tuottaa luvut 25, 50, 7, 3, 9 ja 5.*
>
> *Seuraavaksi opettaja arpoo satunnaislukugeneraattorilla 905.*
>
> *Nyt opiskelijat voivat kokeilla paperilla, laskimella ja/tai taulukko-ohjelmalla kuinka he voisivat päästä haluttuun lopputulokseen.*
>
> *Esimerkin tavoite voidaan saada vaikkapa seuraavasti 25-7 = 18, ja 18 × 50 = 900 ja lopulta 900 + 5 = 905.*
>
> *Pisteytys toimii siten, että oikeasta vastauksesta saa 10 pistettä, 1-5 päähän pääsemistä saa 7 pistettä ja 6-10 päähän oikeasta vastauksesta pääseminen tuottaa 5 pistettä.*
>
> *Peliä pelataan kunnes joku opiskelijoista tavoittaa 30 pistettä.*

Harjoituksen tavoitteena on kokeilla erilaisia loogisia operaatioita, joilla päästään kuitenkin samaan haluttuun lopputulokseen. Haluaisin toteuttaa tämän siten, että opiskelijat voisivat itse käyttää Tie koodariksi -sivuston python editoria laskimena tai jopa mahdollisesti koodata vastauksia itse. Progressio kynästä ja paperista taskulaskimen kautta pythoniin on mielestäni kohtuullisen selkeä.

Arvioinnissa kiinnitetään erityisesti huomiota loogisten operaatioiden osaamiseen, ei niinkään suoraan ohjelmointiosaamiseen. Arviointi on sangen helppoa, sillä pelissä on pisteytysjärjestelmä ja ryhmien työskentelyä pystyy seuraamaan. Ryhmät tekevät myös itsearvioinnin ryhmätyön sujumisesta ja tehtävien onnistumisesta.