Torssonen Taru

OHJELMOINNIN OPETUS PERUSKOULUN 5-6 LUOKILLA JA OHJELMOINNILLISEN AJATTELUN VAIKUTUKSET YLI OPPIAINERAJOJEN

Kandidaatintutkielma
KASVATUSTIETEIDEN TIEDEKUNTA
Laaja-alainen luokanopettajakoulutus
2017

Oulun yliopisto
Kasvatustieteiden tiedekunta
Ohjelmoinnin opetus peruskoulun 5-6 luokilla ja ohjelmoinnillisen ajattelun vaikutukset yli oppiainerajojen (Taru Torssonen)
Kandidaatin tutkielma, 28 sivua
Marraskuu 2017

Syksyllä 2016 astui voimaan uusi perusopetuksen opetussuunnitelma, jonka myötä ohjelmoinnin opetus kirjattiin osaksi suomalaista peruskoulun matematiikan opetusta. Vaikka sisällöt ovat opetussuunnitelmassa matematiikan alla, on niitä tarkoitus integroida myös muihin oppiaineisiin, kuten liikuntaan ja käsitöihin. Tutkimuksen tavoitteena oli tutustua ohjelmoinnin opetukseen ja ohjelmoinnillisen ajattelun merkitykseen osana oppilaan monilukutaitoa ja tieto- ja viestintäteknistä osaamista aiempaa tutkimuskirjallisuutta hyödyntäen.

Perusopetuksen opetussuunnitelman perusteissa ohjelmoinnin opetukseen liitetään algoritmisen ajattelun, eli ongelmanratkaisutaitojen kehittäminen. Ohjelmoinnillisen ajattelun harjoittelulla on havaittu olevan positiivisia vaikutuksia oppilaan ajattelutaitoihin. Sen avulla voidaan harjoittaa ongelmanratkaisutaitoja, joissa ongelma pilkotaan osiin ja pystytään siten ratkaisemaan jopa omaa taitotasoa vaikeampia tehtäviä. Kun tekniikka on opittu hyvin, sitä on osattu soveltaa myös muiden oppiaineiden sisältöihin. Ohjelmoinnin opettamiseen on suunniteltu useita erilaisia sovelluksia ja ympäristöjä, joiden käytössä on hyvä huomioida ikätasot ja se ettei niitä käytetä vain itseisarvona, vaan niille on olemassa oppitunnin kannalta jokin pedagoginen merkitys.

Avainsanat: ohjelmointi, opetus, ohjelmointiympäristöt, peruskoulu, ongelmanratkaisu

SISÄLTÖ

LYHENTEET

IT	Information Technology
OPH	Opetushallitus
OPS	Perusopetuksen opetussuunnitelman perusteet 2016
TVT	Tieto- ja viestintäteknologia

1 JOHDANTO

Kandidaatintyöni aiheena ovat ohjelmoinnin opetus ja sen vaikutukset oppilaan monilukutaitoon ohjelmoinnillisen ajattelun kautta. Joulukuussa 2014 Opetushallitus (OPH) hyväksyi uudet opetussuunnitelman perusteet 2016 (OPS). Opetussuunnitelmatyötä tehtiin useamman työryhmän (Opetushallitus, 2016) voimin vuodesta 2012 alkaen ja uusi OPS otettiin käyttöön syksyllä 2016. Uutena aihealueena mukaan tuli ohjelmoinnin opetus, jota tullaan opettamaan kaikilla peruskoulun luokilla 1-9 (vuoteen 2019 mennessä).

Perusopetuksen opetussuunnitelman perusteissa kantavana teemana on laaja-alainen osaaminen. Se on jaettu seitsemään eri alalajiin: ajattelu ja oppimaan oppiminen, kulttuurinen osaaminen, vuorovaikutus ja ilmaisu, itsestä huolehtiminen ja arjen taidot, monilukutaito, tieto- ja viestintäteknologinen osaaminen, työelämätaidot ja yrittäjyys ja osallistuminen, vaikuttaminen ja kestävän tulevaisuuden rakentaminen (Opetushallitus, 2015). Näistä niin kutsutuista L-taidoista tutustutaan tässä kandidaatin työssä tarkemmin monilukutaitoon ja tieto- ja viestintäteknologiseen osaamiseen kappaleessa kolme. Tieto- ja viestintätekniikan (TVT) rooli opetuskäytössä on ollut pitkään muutoksessa ja ohjelmoinnin opetus muuttaa sen asemaa entisestään kun siihen liittyviä sisältöjä on OPS:sin mukaan tarkoitus integroida useampaan eri oppiaineeseen. TVT:n opetuskäyttöön ovat paremmin perehtyneet Ari Haasio ja Minna Haasio teoksessaan Pulpetit virtuaalivirrassa. Teoksesta löytyy paljon kansainvälistä vertailua, jonka mukaan käyttöastetta tarkasteltaessa Suomen luvut TVT:n opetuskäytön määrissä ovat yllättävän alhaiset, jopa Pohjoismaiden välillä vertaillessa. He nostavat suurimmiksi käytön esteiksi ”riittämättömät tietotekniset taidot, itseluottamuksen puutteen tietotekniikan käytössä, pedagogisen käytön koulutuksen puute, puutteellisen kyvyn tai halun seurata tietoteknistä kehitystä sekä opettajille tarkoitettujen tvt-koulutusohjelmien puutteen” (2008, s. 69). OPS:n uudistuksen myötä kynnys TVT:n opetuskäyttöön on vaarassa nousta entisestään.

Syksyllä 2016 voimaan astuneen opetussuunnitelman myötä ohjelmoinnin opetus kirjattiin osaksi suomalaista peruskoulun matematiikan opetusta. Vaikka ohjelmoinnin tavoitteet on OPS:ssa kirjattu matematiikan sisältöihin, on opetusta tarkoitus integroida myös muihin oppiaineisiin, kuten käsitöihin ja liikuntaan. OPS:ssa (2015) käytetään algoritmisen ajattelun termiä, jolla viitataan ohjelmoinnissa käytettyyn ongelmanratkaisutapaan, jossa tehtävä pilkotaan osiin ja ratkaistaan vaiheittain.

Näkökulma kandidaatintyössäni on peruskoulussa, erityisesti 5-6 luokkalaisten opetuksessa. Ohjelmoinnin opetus nähdään usein irrallisena osana opetusta, eikä osata hahmottaa sen tarjoamia mahdollisuuksia oppilaan L-taitojen, kuten luovuuden ja ongelmanratkaisun tai oppilaan kognitiivisten taitojen eli tiedon käsittelyn ja käyttämisen kehitykseen. Työn tarkoituksena on selvittää, voidaanko laadukasta ohjelmoinnin opetusta tarjoamalla saada aikaiseksi hyviä oppimistuloksia myös muissa oppiaineissa.

Opetussuunnitelman mukaan 1-2 luokkien kohdalla ohjelmoinnin opetus on vielä hyvin pitkälti leikkiin pohjautuvaa, 3-6 luokilla käytetään visuaalisia ohjelmointivälineitä ja vasta yläkoulun puolella aloitetaan perehtyminen johonkin ohjelmointikieleen. Sama jako on nähtävissä myös ohjelmointivälineissä, esimerkiksi Scratchista on olemassa esi- ja alkuopetukseen soveltuva Scratch Jr., kun taas ylemmillä luokilla voidaan hyödyntää Scratchia. Alakouluissa ohjelmoinnin opetusta voidaan soveltaa lähes kaikkiin oppiaineisiin hyödyntäen esimerkiksi toimintaohjeita liikuntatunnilla, edellyttäen kuitenkin, että käyttö on tarkoituksenmukaista eli pedagogisesti merkityksellistä. Yläkoulun puolella ohjelmointi on kirjattu osaksi matematiikan opetusta, mutta sisältöjä löytyy muistakin oppiainesta, kuten teknisestä työstä. Esi- ja alkuopetuksessa ohjelmointi on tarkoitus aloittaa toimintaohjeisiin perustuvilla leikeillä. Osaamistavoitteisiin kuuluu, että alakoulun päättyessä, saadakseen hyvän arvosanan, oppilaan tulee osata ohjelmoida toimiva ohjelma graafisessa ohjelmointiympäristössä (Opetushallitus, 2015). Opetussuunnitelmassa ohjelmoinnin opetukseen varattu aika on niin vähäinen, että tavoitteen saavuttaminen vaatii paljon työtä ja oppiaineiden kesken tapahtuvaa integraatiota sekä monialaisten kokonaisuuksien hahmottamista, varsinkin jos sen sisältö aiotaan täysin oppituntien aikana saavuttaa.

Perusopetuksen opetussuunnitelman perusteissa ohjelmoinnin sisältö on määritetty hyvin pintapuolisesti, se ei ota kantaa edes opetuksessa käytettävään ohjelmointikieleen tai -välineisiin. Kuitenkin opettajan toimintaa ohjaavia dokumentteja voi löytyä kunnankin tasolla useita. Kuitenkin ohjelmoinnin opetuksen painopiste on algoritmisessa ajattelussa, eikä itse koodaamisessa. Koodaaminen nähdään yhtenä työvälineenä ongelmanratkaisussa, sen avulla oppimisesta tehdään mielekkäämpää. Teknologian opetuskäyttö nähdään usein motivaatiota lisäävänä. Järvelä, Häkkinen ja Lehtinen (2006, s. 93) toteavat, että ”oikein käytettynä teknologian avulla voidaan tukea oppijoiden itsesäätelyä ja tuoda oppimiseen erilaisia elementtejä, jotka ylläpitävät motivaatiota ja kiinnostusta”. Opettajan vastuulle jää keksiä tavat, joilla oppimista tuetaan edellä mainituilla keinoilla.

Tässä kandidaatin työssäni perehdyn ohjelmoinnin opetukseen ja tieteellisin perusteluihin joiden perusteella voidaan perustella ohjelmoinnin merkitys osana valtakunnallista opetussuunnitelmaa. Opetussuunnitelman perusteiden uudistuksen myötä aiempaa suuremman roolin saaneeseen ohjelmoinnin opetukseen ei ole mitään lähtötasoa tai pätevyysvaatimusta. Koodi2016-opas näkee tämän vaihtelevuuden positiivisena asiana, sillä se opettaa erilaista osaamista ja kokeilemisen kulttuuria (Liukas & Mykkänen, 2014). Esi- ja alkuopetuksessa opettajan lähtötason merkitys ei ole kovin suuri, sillä ohjelmoinnin näkökulmasta opetettavat sisällöt ovat kohtalaisen yksinkertaisia.. 3-6 luokilla erot alkavat jo näkyä. Miten voidaan taata kaikille oppilaille yhtäläiset mahdollisuudet opiskella ohjelmointia ja ohjelmallista ajattelua riippumatta opettajasta, koulusta tai kunnasta? Jos opettaja ei ole koskaan koodannut, eikä siihen erityisemmin halua edes perehtyä, voi jopa samasta koulusta löytyä merkittävät erot oppilaiden osaamisessa, jotka näkyvät viimeistään yläkouluun siirryttäessä. Tämän tutkimuksen tavoitteena on selkeyttää ohjelmoinnin merkitystä suhteessa oppilaan oppimistaitoihin, kuten ongelmanratkaisukykyyn, ja sen myötä korostaa sen mahdollisia positiivisia vaikutuksia yli oppiainerajojen. Ohjelmoinnin opetusta käsiteltäessä, on tärkeätä tutustua myös aiemmin mainittuun ohjelmoinnilliseen ajatteluun tarkemmin. Esimerkiksi siihen, voiko ohjelmoinnillisella ajattelulla olla vaikutuksia yli oppiainerajojen? Tai miten ohjelmoinnillista ajattelua voidaan soveltaa ja mitä on monilukutaito ja miten se liittyy ohjelmointiin?

Opettajien rooli on erittäin merkittävä ohjelmoinnin oppimisessa. Lukuisista muista maista poiketen, opetussuunnitelman perusteiden toteutumista seurataan Suomessa vain oppimistavoitteiden täyttymisen myötä, varsinaista tarkkailua ei tehdä. Koulujen luotetaan kantavan vastuunsa oppimisesta ja opetuksesta. Haasteeksi nousevat olemassa olevat asenteet, sillä kaikki opettajat eivät koe ohjelmoinnin opetusta mielekkääksi ja ohittavat sen opetuksen mahdollisimman vähällä. Opettajien taitoerot näkyvät myös tutkimustuloksissa: jopa yhden vuoden opetuksen jälkeen voidaan havaita merkittäviä eroja saman luokka-asteen oppilaiden osaamisessa opettajan taitotason takia (Sanger;Willson;Davies;& Whittaker, 1997). Tämä luo merkittäviä tasoeroja pidemmällä aikavälillä tarkasteltaessa ja korostuu erityisesti ryhmien muuttuessa, esimerkiksi nivelvaiheessa, oppilaiden siirtyessä yläkoulun puolelle. Jos täydennyskoulutusta eikä välineistöä ole riittävästi, miten varmistetaan kaikille oppilaille yhtäläiset mahdollisuudet ohjelmoinnin sisältöjen oppimiseen?

Syksyllä 2015 yli 2000 opettajaa osallistui Aalto-yliopiston ja It-kouluttajat ry:n järjestämällä verkkokurssille ”Koodiaapinen” (Lehtinen, 2015). Vaikka määrä on suuri, se kattaa vain murto-

osan opettajista, jotka ovat joutuneet ohjelmointia syksystä 2016 alkaen opettamaan. Esi- ja alkuopetuksen puolella suuren osan ohjelmoinnin opetuksesta voi hoitaa oppikirjojen avulla, mutta 3-6 luokkien opettajilla täytyy olla käsitys ohjelmoinnista ja siitä miten ohjelmointiin liittyviä sisältöjä voidaan opettaa. Vastuu oman osaamisen päivittämisestä on niin opettajalla itsellään, koulun rehtorilla kuin paikallisella opetustoimellakin.. Täytyy toivoa, että opettajat osaavat hyödyntää kollegoidensa taitoja, sillä ohjelmointia valmiiksi osaavia opettajia on vähän. Tämän lisäksi suomenkielisen materiaalin puute aiheuttaa haasteita. Opettajat ovat tehneet osan materiaaleista talkootyönä, kuten esimerkiksi Koodiaapinen. Haasio & Haasio korostavat sosiaalisen median yhteisöjen merkitystä. Vaikka verkkoyhteisöjä pidetään usein viihteellisinä, voi niiden avulla verkostoitua myös ammatillisesti ja saada tukea oman opettamisen tueksi. Tällaisissa yhteisöissä jaetuille mielenkiinnonkohteille voi luoda helposti oman alustan, jossa aiheesta kiinnostuneet pystyvät kommunikoimaan keskenään (Haasio & Haasio, 2008). Opettajat ovat luoneet syksyyn 2017 mennessä lukuisia verkkoyhteisöjä ohjelmoinnin opettamiseen liittyen, esimerkiksi Koodausta kouluun-ryhmä. Siellä aiheesta kiinnostuneet vinkkaavat toisilleen hyväksi koettuja käytänteitä, nettisivuja, joista löytyy tarvittavia tietoja ja apuakin saa kysyä.

Asenteet ohjelmointia kohtaan vaikuttavat merkittävästi ohjelmoinnin opettamiseen. Lähtökohdista huonoin on se, jossa opettajaa ei ole itse kiinnostunut ohjelmoinnin opettamisesta. Tästä huolimatta opettajan velvollisuus on tarjota jokaiselle oppilaalle opetusta myös ohjelmoinnista. Ohjelmointia voi opettaa niin monella tapaa, että myös vähemmän kiinnostuneille on tarjolla soveltuvia keinoja. Negatiivista asennetta ei helpota se, että useiden ohjelmointia epäilevien opettajien täytyy asettua taas oppilaan rooliin uuden aiheen kanssa. Vaikka on olemassa useita teknologiaympäristöjä oppimisen tueksi, mutta silti oppiakseen opettajan täytyy löytää oma motivaatio ja ponnistella kehittääkseen omia tietojaan ja taitojaan (Järvelä;Häkkinen;& Lehtinen, 2006). Vaikka ympäristöt ovat oppimisprosessia tukevia, kukaan ei voi opettajan puolesta ohjelmoinnin oppimista tehdä.

Ohjelmoinnin opetuksen suhteen on olemassa vielä paljon haasteita. Aiemmin mainittujen resurssien, materiaalien ja asenteiden lisäksi tällä hetkellä ”puuttuu pedagoginen ymmärryksen puute, ei tiedetä pitäisikö aloittaa tekemisellä vai teorialla” (Liukas & Mykkänen, 2014, s. 62). Varsinkin esi- ja alkuopetuksessa teoria tulee vielä mukavasti tekemisen rinnalla, kun harjoitusten aloittamiseen ei vaadita kovin tarkkaa ohjelmoinnin osaamista. 3-6 luokilla pitäisi olla jo jonkinlainen pohjatieto ohjelmoinnillisesta ajattelusta ja syventävää tietoa on hyvä rakentaa

tekemisen tueksi. Yläkoulussa rakennetta voi ohjelmointikielten opettelun myötä muuttaa teorialähtöiseksi siten, että ohjelmointiin liittyviä harjoituksia voidaan pohjustaa aihetta käsittelevillä teoriatunneilla.

Aiheen valintaan vaikutti myös oma kiinnostus ohjelmoinnin opetukseen. Olen aina pitänyt tieto- ja viestintätekniikkaa tärkeässä roolissa opetuksen apuvälineenä ja opiskelen sivuaineena ohjelmointia tietojenkäsittelytieteiden laitoksella. Opetussuunnitelmauudistus herätti paljon keskustelua ja huolta luokanopettajaopiskelijoiden keskuudessa, sillä omat taidot ja osaaminen koettiin paikoitellen riittämättömäksi tämän uuden aiheen edessä. Ohjelmoinnin osuus OPS:ssa on vähäinen, koska sen opetus ei tullut omaksi oppiaineekseen vaan on osana muiden oppiaineiden sisältöjä. Sen vuoksi vaarana on ohjelmoinnin jääminen täysin muiden sisältöjen jalkoihin, ellei opettajalla omaa kiinnostusta aiheeseen ole.

Tämä tutkimus on toteutettu kirjallisuuskatsauksena. Tavoitteena on kuvata ohjelmoinnin opettamisen merkitystä useasta näkökulmasta. Luvussa 2 ovat tutkimuskysymykset, luvussa 3 tutustutaan tieto- ja viestintäteknologian opetuskäytön muutokseen. Luvussa 4 tarkastellaan ohjelmoinnin opettamisen merkitystä laajemmassa mittakaavassa, esimerkiksi mitä tarkoittaa ohjelmoinnillinen ajattelu ja miten sitä voidaan soveltaa. Lisäksi tutkitaan onko ohjelmoinnillisella ajattelulla jotain kognitiivisia vaikutuksia ja mitä on monilukutaito. Luvut 5 ja 6 käsittelevät ohjelmoinnin konkreettisempaa opetusta, miten sitä voidaan toteuttaa niin visuaalisissa ohjelmointiympäristöissä kuin koneettomasti. Luvussa 7 ovat johtopäätökset ja yleistä pohdintaa aiheesta.

2 TUTKIMUKSEN TAVOITE JA TUTKIMUSKYSYMYKSET

Tutkimuksen tavoitteena on tutustua ohjelmoinnin opetukseen ja ohjelmoinnillisen ajattelun merkitykseen osana oppilaan monilukutaitoa ja tieto- ja viestintäteknistä osaamista aiempaa tutkimuskirjallisuutta hyödyntäen. Päätutkimuskysymykset ovat:

- Mitä on ohjelmoinnillinen ajattelu peruskoulukontekstissa ja miten sitä opetetaan?
- Mitä ohjelmointi on peruskoulukontekstissa?

3 TIETO- JA VIESTINTÄTEKNOLOGIAN ASEMA MUUTOKSESSA

Ohjelmoinnin merkitys on kasvanut huomattavasti digitalisaation myötä. Käytämme kaikkialla läsnä olevaa ja arkeemme sulautunutta teknologiaa jatkuvasti ja keskeistä tälle teknologialle on se, että ilman ohjelmistoja ne eivät tekisi niitä toimintoja joita varten ne ovat suunniteltu. Arkemme on siis jo nyt digitilasoinutta monella tapaa. Taru Rastas (viestintäneuvos, Liikenne- ja viestintäministeriö) on todennut digitalisaation ja tulleen huomaamatta osaksi arkielämää, sitä tarvitaan muun muassa kaupoissa, sairaaloissa ja julkisessa liikenteessä (Liukas & Mykkänen, 2014). Ohjelmointia opetetaan peruskouluissa jo useammassa maassa. Sitä pidetään tulevaisuuden kannalta keskeisenä taitona ja siksi Suomen täytyy pysyä kehityksessä mukana. Tulevaisuudessa nähdään ohjelmoijia tarvittavan vielä nykyistä enemmän kaiken automatisoituessa, lisäksi ettei ohjelmoinnin ulkoistamisen ole niin helppoa kuin on aiemmin luultu (Liukas & Mykkänen, 2014). Liukkaan ja Mykkäsen mielestä ohjelmointi on ollut aiemmin hyvin rajatun joukon harrastus ja sillä on nyt OPS:n uudistuksen myötä mahdollisuus demokratisoitua todella. He korostavat ohjelmointitaidon merkitystä vaikkei koskaan työkseen koodaisi, sillä suuri osa nykyajan toiminnoista vaatii koodia ja jonkinlainen näkemys sen rajoituksista ja mahdollisuuksista helpottaa monenlaisessa yhteistyössä. Lisäksi heidän toiveenaan on, että ohjelmoinnin opetus lisää tyttöjen määrää tulevaisuuden koodaajina, ala on tällä hetkellä varsin miespainotteinen. (Liukas & Mykkänen, 2014)

Howard Rheingold (2003) kirjoittaa kirjassaan *Mobiilijoukot* aina vain virtualisoituvasta maailmasta, kirja tukee hyvin ajatusta siitä kuinka ohjelmointi on tärkeä osa tulevaisuutta ja siksi sen opettelu on hyvä aloittaa mahdollisimman nuorena. Vuonna 2003 kirjoitetussa kirjassaan hän ennustaa tieto- ja viestintäteknologian kehityssuuntia ja kertoo vierailleensa IBM:n ja MIT:n tiloissa, virtuaalisesti (Rheingold, 2003, s. 94). Mobilisoituva maailma antaa opettajalle rajattomia mahdollisuuksia tuntisuunnitelmien suhteen. Teknologian kehittyessä laitteiden hinnat ja saatavuus vaihtelevat ja lisääntyvät, kohta tavallisesta suomalaisesta luokkahuoneesta on mahdollista käydä vierailulla vaikka Googlen toimistolla Yhdysvalloissa, poistumatta lainkaan turvallisen luokkahuoneen sisältä. Opettajan tulee pysyä mukana teknologian kehityksessä ja opetella käyttämään kaikkia sen mukana saapuvia mahdollisuuksia hyödyntääkseen niitä opetuksessa parhaalla mahdollisella tavalla.

Teknologian kehityksestä huolimatta, TVT:n opetuskäytössä on edelleen haasteita. Vuonna 1997 Britanniassa julkaistiin kahden vuoden seurantatutkimuksen tulokset 4-9 vuotiaiden lasten tieto- ja viestintätekniikan käytöstä niin kouluissa kuin vapaa-ajalla. Yhtenä käsiteltävänä

aiheena oli opettajien taitotaso IT:n opetuskäytössä. Tutkimuksessa oli huomattu usein tilanteita, joissa oppilaan taidot olivat paremmat kuin opettajan. Harvassa luokkahuoneessa TVT:tä osattiin hyödyntää opetuksen tukena, yleisempää oli TVT:n erillinen, rajattu opetus- ja käyttöaika (Sanger;Willson;Davies;& Whittaker, 1997). Usein oppilaat vain kirjoittivat tietokoneella puhtaaksi aiemmin käsinkirjoitettua tekstiä. 20 vuoden aikana, kehitys on ollut nopeaa ja lukuisat sovellukset-ja välineet ovat tulleet opettamisen ja oppimisen tueksi. Suomessa on kuitenkin vielä lukuisia oppilaitoksia joiden varustelu on puutteellista ja opettajien osaamisessa on puutteita. Myös asenteet ovat paikoitellen sellaisia että TVT on vain ”välttämätön paha” joka on vain hoidettava mahdollisimman pian alta pois.

Yhtenä haasteena TVT:n opetuskäytön suhteen voidaan pitää koulujen ja luokkahuoneiden kokoa. Suomalaisten peruskoulujen määrä on ollut useamman vuoden laskussa. Vuonna 2015 Suomessa on 1709 vuosiluokkien 1-6 peruskoulua (vrt. 2005 2660kpl) (Tilastokeskus, 2016). Ero suurimpien ja pienimpien koulujen oppilasmäärissä on valtava. Vaikka pienryhmien oppilasmäärät pyritään pitämään samoina koulun koosta riippumatta, on sekä pienillä että suurilla kouluilla omat heikkoutensa ja vahvuutensa. Ohjelmoinnin opetukseen liittyen resurssien erot saattavat olla merkittävät. Pienellä koululla on usein pienempi budjetti myös laitehankintoihin, mutta opettajalla taas on usein paremmin aikaa keskittyä yksittäiseen oppilaaseen. Isoissa kouluissa taas opettajakunnasta löytyy monen alan asiantuntijoita, pienessä koulussa yksi ainoa opettaja opettaa kaikki aineet, eikä niistä jokainen voi edes olla sitä ominta aluetta.

Liukkaan ja Mykkäsen kirjoittamassa Koodi2016 oppaassa todetaan TVT:n tilan olleen heikko, tietotekniikan valinnaisena valinneiden oppilaiden määrä vähentyi 10 prosenttia välillä 2008–2010 (Liukas & Mykkänen, 2014, s. 51). Oppaan mukaan tähän vaikutti vahvasti se, ettei tietotekniikan opetusta oltu uudistettu viimeiseen kymmeneen vuoteen. Pitkään on ollut vallalla uskomus että oppilaat oppisivat tarvittavat taidot vapaa-ajallaan, myöskin vanhentunut opetustarjonta ei ole tarjonnut riittävästi haastetta eikä kurssien sisällöissä ole ollut tärkeää mielekkyyttä. Liukkaan ja Mykkäsen (2014) mukaan kysyntää kuitenkin olisi, sillä oppilaat toivovat lisää TVT:tä kouluihin. Tällä toiveella saatetaan myös viitata tarpeeseen uudistaa opetusta ja oppimista yleisimminkin. Muutos on kuitenkin hidas ja se vaatii rohkeutta myös opettajalta. Yhdeksänkymmentäluvun aikana tuli tärkeä muutos, jonka myötä TVT:n opetus muuttui teknisestä inhimillisemmäksi (Tella;Vahtivuori;Vuorento;Wager;& Oksanen, 2001). TVT on haluttu OPS:ssa nostaa erilliseksi L-taidoksi vaikka se olisi voinut olla esimerkiksi monilukutaidon osa. Tätä inhimillisempää TVT:tä kuvastaa nykypäivänä hyvin myös TVT:n integrointi

kaikkiin aineisiin, sitä ei nähdä enää omana aineenaan vaan hyödynnetään sen koko kirjoa aina sosiaalisesta mediasta käsillä olevaan ohjelmointiin.

”Monilukutaito liittyy erilaisten tekstien tulkitsemiseen, tuottamiseen ja arvottamiseen, kykyyn hankkia, muokata, esittää ja arvioida tietoa, rakentaa identiteettiä ja kriittistä ajattelua sekä oppimista ja eettistä pohdintaa monikulttuurisessa ja kulttuurisesti moninaisessa maailmassa” (Kupiainen;Kulju;& Mäkinen, 2015, s. 16). Teknologian ja muun kehityksen myötä monilukutaitoon liitetään myös mm. käsitteet teknologinen lukutaito, informaatiolukutaito, verkkolukutaito, tietokonelukutaito ja medialukutaito. (Kupiainen;Kulju;& Mäkinen, 2015) Monilukutaitoon kerrostuu siis useita oppiaineita, joiden voidaan nähdä linkittyvän toisiinsa. Monilukutaidon osa-alueet kerryttävät toistensa osaamista rinnakkain, esimerkiksi ohjelmoinnin opettelulla voidaan selkeästi kerryttää lukutaitoa myös matematiikan tai kotitalouden oppiainesisällöissä, tai kuvaamataidon sisällöillä voidaan harjoitella kulttuurin lukutaitoa. Perusopetuksen opetussuunnitelman perusteissa on eritelty laaja-alaisia osaamistavoitteita ja monilukutaito on yksi näistä. Osana monilukutaitoa nähdään että ”ihmisenä kasvaminen, opiskelu, työnteko sekä kansalaisena toimiminen nyt ja tulevaisuudessa edellyttävät tiedon- ja taidonalat ylittävää ja yhdistävää osaamista” (Opetushallitus, 2015, s. 20). OPS:n laaja-alaisissa osaamistavoitteissa on myös tieto- ja viestintäteknologinen osaaminen ja olennainen osa sitä on mainittu juurikin monilukutaito.

4 OHJELMOINNILLINEN AJATTELU

”Computational thinking eli ohjelmoinnillinen ajattelu viittaa ongelmien ajatteluun siten, että tietokone pystyy ratkaisemaan ne. Siihen kuuluvat looginen ajattelu, kyky tunnistaa yhteneväisyyksiä, algoritmien luominen, ongelman purkaminen osiin ja sen yleistäminen.” (Liukas, 2015, s. 110) Ohjelmoinnilliselle ajattelulle on useampia määritelmiä, mutta pääpiirteet pysyvät lähes samana ja lähes poikkeuksetta algoritminen ajattelu on yksi niistä (Denning, 2017). Myös perusopetuksen opetussuunnitelman perusteiden osaamistavoitteissa ohjelmoinnin opetus ja algoritmisen ajattelun kehittäminen kytketään toisiinsa (Opetushallitus, 2015). Ohjelmoinnillista ajattelua ja algoritmista ajattelua pidetäänkin usein lähes synonyymeina. Algoritmit ovat osa arkista elämää, esimerkiksi reseptit tai ajo-ohjeet ovat tällaisia (Snyder, 2006). Harjoittelemme siis algoritmista ajattelua ilman suurempaa tiedostusta asiasta.

Toisin kuin usein ajatellaan, ohjelmoinnillisella ajattelulla tarkoitetaan tapaa jolla ihmiset ratkaisevat ongelmia, ei sitä että ihmisten tulisi ajatella kuin tietokoneet (Wing, 2006). Wing myös toivoi, että ohjelmoinnillisen ajattelun opetus popularisoituisi kunnolla, eikä olisi saatavilla vain rajatulle joukolle alan ammattilaisia (2006). Kymmenen vuotta myöhemmin, Wing toteaa nähneensä vuonna 2006 tietojenkäsittelytieteen mahdollisuudet huomattavasti positiivisemmin kuin kollegansa, josta polveutuikin hänen toiveensa ohjelmoinnillisen ajattelun opettamisesta. Wing ei kuitenkaan uskonut, että ohjelmoinnillisen ajattelun asema voisi saavuttaa paikan opetussuunnitelmassa hänen elinaikanaan ja onkin hyvin ilahtunut muutoksesta. Wing ennustaa, että tulevaisuudessa ohjelmoinnillisesta ajattelusta tulee vielä olennainen osa elämäämme, kirjoittamisen ja lukemisen rinnalle. (2016)

Peter Denning (2017) tarkastelee artikkelissaan ohjelmoinnillisen ajattelun haasteita. Suurimpina ongelmina hän pitää lukusia, jopa hieman epätarkkoja määritelmiä ohjelmoinnillisen ajattelun sisällöistä, epäselvyyttä siitä kuinka ohjelmoinnillista ajattelua voidaan testata ja väitettä jonka mukaan ohjelmoinnillisesta ajattelusta on hyötyä kaikille. (Denning, 2017) Empiiristen tutkimustulosten vähäisyys on tietysti haaste, luotettavia tutkimustuloksia ei ainakaan vielä ole saatavilla. Denning (2017) on myös oikeassa siinä, että opetusalan ammattilaiset ovat käyttäneet useita erilaisia opetuksen ajattelumalleja aikojen saatossa, löytyy esimerkiksi niin kriittistä kuin loogistakin ajattelua korostavaa opetustyyliä. Mikään ei kuitenkaan poissulje ohjelmoinnillisen ajattelun mahdollisuuksia. Kaikille Denningin mainitsemille ajattelutavoille on ollut aikansa ja paikkansa, ehkä nyt on vuorossa ohjelmoinnillisen ajattelun aika. Ohjelmoinnillinen

ajattelu on kuitenkin vakiinnuttanut asemaansa jo vuosikymmenten ajan ja se sisältää piirteitä useammasta aiemmastakin opetuksen ajattelumallista.

Denning (2017) kyseenalaistaa voidaanko ohjelmoinnillisesta ajattelusta todella saada hyötyä esimerkiksi taiteilijoille ja asianajajille. Wing (2016) toteaa, että tutkijat kaikilla aloilla, niin humanistisissä kuin yhteiskuntatieteissäkin löytävät uutta tietoa ohjelmoinnillisilla välineillä ja tavoilla. Lisäksi jäljempänä todistetaan algoritmisen ajattelun kehittävän ongelmanratkaisutaitoja ja on tärkeää huomata, että vaikka kyseisiä taitoja ei työelämässä koskaan tarvitsisi, ei voida väittää ettei kyseisillä taidoilla olisi merkitystä arkielämässä.

4.1 Ohjelmoinnin opiskelun vaikutukset laajemmassa mittakaavassa

Kognitiivisilla perustaidoilla viitataan tiedon saamiseen, käsittelyyn ja käyttöön liittyviin prosesseihin eli tietämistoimintoihin. ”Hahmopsykologia selittää tiettyjen elementtien yhteenkuuluvuutta, sitä miten ymmärrämme kokonaisuutta, jollaiseksi havaintomme ovat ohjelmoituneet: Näemme, kuulemme ja ymmärrämme asioita hahmolakien mukaisesti.” (Haapasalo, 1994) Ohjelmoinnillisen ajattelun kehittyminen harjoittaa myös kognitiivisia taitoja, kun ongelmia opitaan pilkkomaan osiin ongelmanratkaisua varten. Täten ohjelmoinnillisen ajattelun myötä voidaan hahmottaa ongelmien vaiheita ja ratkaisuja helpommin, myös muissa oppiaineissa. Ongelmanratkaisutaitojen kehittyminen ei rajoitu vai käsitteillä olevaan aineeseen, vaan kun taidot kehittyvät, voidaan niitä käyttää myös muissa oppiaineissa. Ongelman purkaminen osiin ratkaisua varten on hyödyllinen taito myös arkielämässä.

Metakognition mahdollisuuksia teknologisissa oppimisympäristöissä on tutkittu useaan otteeseen. Teknologisessa oppimisympäristössä oppilaille voidaan tarjota tukea oman oppimisensa reflektointiin ja opintojen itseohjautuvuuteen. Ohjelmoinnin opetuksessa voidaan hyödyntää näitä taitoja oppimisympäristön vaikeustason ja etenemisvauhdin muutoksilla. Näiden tavoitteena on rohkaista oppilaita ratkaisemaan tehtäviä, joiden vaativuustasoon oppilas ei vielä muuten yllä. Tätä kutsutaan oppilaiden metakognition tueksi ja tutkimustulosten mukaan oppilaat ovat onnistuneet tällaisella mallilla ongelmanratkaisuun, joka olisi ollut mahdotonta täysin itsenäisesti. (Järvelä;Häkkinen;& Lehtinen, 2006)

Ohjelmoinnin vaikutuksia oppilaiden kognitiivisiin taitoihin pohdittiin jo 1984. Ohjelmointiympäristön parhaana antina pidettiin sitä, että sen sijaan että oppilaille opetettaisiin että mitä pitää ajatella, ohjelmoinnin opetus painottuu opettamaan miten se tehdään (Lochhead & Clement, 1979 teoksessa (Clements & Gull, 1984)). Tällä mallilla havaittiin olevan positiivinen

vaikutus oppilaan kehitykseen, oppilaat pystyivät lyhyessä ajassa hyödyntämään oppimiaan taitoja ja ratkaisemaan omaa taitotasoaan vaativampia tehtäviä. (Clements & Gull, 1984) Havainnot tukevat Järvelän, Heikkisen ja Lehtisen (2006) tuloksia. Ajattelumallia ja opittua ongelmanratkaisutaitoa voidaan hyödyntää myös muilla oppitunneilla, sillä kun harjoitusta on saatu tarpeeksi, voivat oppilaat ohjautua jopa itsenäisesti tällaiseen työskentelymalliin.

4.2 Ohjelmoinnillisen ajattelun soveltaminen

Leppäaho (2007) toteaa, että opetussuunnitelmassa ongelmanratkaisua pidetään erittäin tärkeänä osana matematiikan opetusta. Ongelmanratkaisutaitoa pidetään tärkeänä matematiikassa ja se on keskeisimmässä osassa myös ohjelmoinnissa. Ongelmanratkaisutaidon nähdään parantavan ”yleisiä tiedollisia valmiuksia, edistävän luovuuden kehittymistä, olevan osa matemaattista soveltamista ja motivoivan oppilaita matematiikan opiskeluun (Pehkonen, Pekama ja Seppälä, 1991)” (Leppäaho, 2007, s. 17). Ongelmanratkaisutaitoja kehittäessä harjoitusta saadaan myös olennaista osaa, ongelmanratkaisusitkeyttä, koskien: on tärkeää toistaa harjoitusta, vaikka ratkaisua ei ensimmäisellä yrityksellä löytyisikään (Leppäaho, 2007). Tämä sama sitkeys on tarpeen myös ohjelmoinnin opettelussa, varsinkin aloittelija saattaa joutua pyörittämään yhtä ja samaa koodin pätkää pidemmänkin aikaa, löytääkseen oikean, toimivan ratkaisun.

Seymour Papert on pyrkinyt opettamaan lapsille ongelmanratkaisutaitoja, jotka ovat hyödynnettävissä yli oppiainerajojen. Välineeksi hän on valinnut ohjelmoinnin, sillä sen nähdään olevan muodollista ongelmanratkaisua, ongelmana ollen mitä haluat robotin tekevän ja ratkaisuna miten onnistut siinä. Ongelmat hankaloituvat asteittain ja kokeilun avulla oppilas oppii pilkkomaan ongelman osiin ja lopulta ratkaisemaan sen, mikä onkin ongelmanratkaisun keskeisin tehtävä. Tätä mallia hyödynnetään nykyään laajasti opetuksessa. (Alessi & Trollip, 1985)

5 OHJELMOINNIN JA OHJELMALLISEN AJATTELUN OPETTAMINEN JA OPPIMINEN

Ohjelmoinnin opettamiseen ja oppimiseen saatavilla olevien sovellusten- ja välineiden kirjo on laaja. Mielekkäiden välineiden ja materiaalien avulla voidaan lisätä oppilaan motivaatiota. Opetusmateriaalien ja -sovellusten tulee kuitenkin olla pedagogisia ja niiden valinnassa täytyy olla tarkkana. Mikä tahansa tieto- ja viestintätekninen sovellus ei tue pedagogista käyttöä, vaan opettajan tulee tarkastella sitä kriittisesti suunnittelemaansa opetustilanteeseen peilaten eikä pitää vain itseisarvona (Tella;Vahtivuori;Vuorento;Wager;& Oksanen, 2001). Esimerkkinä voidaan pitää aiemmin mainittua mallia, jossa oppilaat vain kirjoittavat koneella puhtaaksi aiemmin tuotettua tekstiä. Paremmin tietotekniikkaa hyödynnetään, jos oppilaat saavat käyttää tietotekniikkaa luomisen apuvälineenä, internetistä voidaan etsiä sisältöä ja tukea tarinalle, tekstinkäsittelyohjelmilla voidaan pyöritellä ja hioa tekstiä paremmaksi, vertaisarvioinnilla voidaan kehittää oppilaiden reflektointi- ja yhteistyötaitoja, TVT:n avulla voidaan toteuttaa vaikka koko kirjoitusprosessi ryhmässä. Samanlaisia osia voidaan harjoitella myös ohjelmoinnillisen ajattelun yhteydessä, sillä samat vaiheet löytyvät ohjelmoinnista. Oppilaat voivat remiksata (eli yhdistää olemassa olevaa koodia omaansa), tarkistaa muiden koodia tai luoda sitä yhdessä ja vaan mahdollisten virheiden sattuessa koodia tutkitaan (debugataan).

Matematiikka voidaan erityisesti ohjelmoinnin myötä nähdä luovan ilmaisun välineenä, kun oppilaat pääsevät kirjoittamaan omaa koodiaan ja luomaan sillä jotain uutta esimerkiksi ohjelmistoja. Opetuskäyttöön soveltuvia palveluita on olemassa useita ja tämä voidaan mahdollistaa esimerkiksi Scratchin tai Scratch Jr:n avulla. Scratchin käytöstä on tehty useita tutkimuksia. Uudessa Seelannissa noin kymmenvuotiaat oppilaat hyödynsivät ohjelmoinnin opettelussa Scratchia ja älytauluja (Forret;Harlow;& Taylor, 2010). Tutkimuksen tuloksissa havaittiin Scratchin tarjoavan hyviä mahdollisuuksia yhteistyölle niin luokkahuoneessa kuin sen ulkopuolellakin. Espanjassa tehdyn kaksivuotisen seurantatutkimuksen tuloksissa havaittiin samoja positiivisia yhteistyön mahdollisuuksia Scratchin tarjoamissa ympäristöissä, lisäksi he havaitsivat visuaalisten työkalujen tehneen oppitunneista hauskempia ja lisäävän oppilaiden motivaatiota, innokkuutta ja sitoutumista (Saez-Lopez;Roman-Gonzales;& Vazquez-Cano, 2016). Visuaalisten ohjelmointiympäristöjen käytössä tulee kuitenkin muistaa aiemmin mainittu pedagoginen käyttö, jotta saavutetaan toivottuja oppimistuloksia. Jokaisella oppilaalla tulee olla mahdollisuus nähdä mitä kaikkea ohjelmoimalla voidaan saada aikaiseksi.

Ohjelmoinnin opetus muuttaa pelaamisen roolia ja käsitystä luokkahuoneissa. Aiemmin pelaaminen nähtiin itseisarvona, keinona kuluttaa aikaa ja viihdyttää oppilaita, se on usein ollut myös palkitsemisen väline. Ohjelmoinnin myötä oppilaat alkavat itse luoda sisältöä pelien maailmassa. Opettajan tulee tarkkailla oppilaiden luomaa sisältöä. Britanniassa tehdyssä tutkimuksessa huomattiin että lapsia ja nuoria usein kiinnostavat groteskit tarinat, tämä selittää esimerkiksi Roahl Dahlin kirjojen menestystä (Sanger;Willson;Davies;& Whittaker, 1997). Monen videopelin sisältö on väkivaltaista. Opettajan tulee tarjota alusta, jolla sisällön tuottaminen on mielekästä, mutta mahdollisuuksia liian väkivaltaisiin sisältöihin ei ole. Toki vaaleanpunaisista poneistakin saa jännittävän pelin tai tarinan, jos oikea kertoja osuu kohdalle.

Kouluissa tuntijako on pysynyt pitkään pääpiirteiltään samana. Nyt kuitenkin mukaan tulee täysin uusi opetettava aihe, joka ei kuitenkaan ole itsenäinen oppiaineensa. OPS:ssa ohjelmointia on kirjattu erityisesti osaksi matematiikan ja käsitöiden opetusta ja yläkoulun puolella ohjelmoinnin opetus on matematiikan opettajien vastuulla. Ohjelmoinnin opettaminen osana matematiikan täyttä tuntisuunnitelmaa on haastavaa. Joitain kevennyksiä on kuitenkin tulossa, sillä esimerkiksi jakokulman opettaminen jää jatkossa kokonaan pois OPS:sta. Opettajan luovuus pääsee ideoinnissa valloilleen ja varsinkin esi- ja alkuopetuksen puolella mahdollisuuksia löytyy. Ohjelmointiin liittyviä komentoharjoituksia on hauska kokeilla esimerkiksi liikuntatunneilla. Käsityötunneillakin harjoituksia ja sovelluksia saa helposti yhdistettyä jo olemassa oleviin tuntisuunnitelmiin. Oppiainesisältöjä tulee integroida muihin oppiaineisiin, varsinkin ohjelmoinnissa, jossa omaa viikkotuntimäärää ei juurikaan ole. Haasteen tähän tuovat opettajien ymmärrys integroitavista oppiainesisällöistä: jos aiheita ei tunneta riittävän hyvin, saattaa jäädä loistavia integrointimahdollisuuksia väliin (Leppäaho, 2007).

Ohjelmointisovelluksilla ja välineillä ei 1-2 luokilla ole vielä niin suurta roolia kun itse opetus on leikkiin pohjautuvaa, OPS:ssa (2015) tavoitteisiin on kirjattu tutustuminen ohjelmoinnin alkeisiin toimintaohjeiden kirjoittamisella ja testaamisella. Opettajan täytyy vain olla kekseliäs, pelkällä maalarinteipillä voi saada ihmeitä aikaan. 3-6 luokilla välineiden merkitys kasvaa, kun opetus on tarkoitus siirtää graafiseen ohjelmointiympäristöön, OPS:n (2015) mukainen tavoite on suunnitella ja toteuttaa ohjelmia graafisissa ohjelmointiympäristöissä. Ohjelmointia pystyy harjoittelemaan pareittain, mutta jokaisella tulisi olla mahdollisuus päästä kokeilemaan koodaamista itse. Omien laitteiden kanssa pelaamisessa on aina omat riskinsä, oppilailla on harvoin

käytössään täysin samantasoiset laitteet. Tämä avaa ikkunan kiusaamisen kohteeksi joutumisesta ja vaatii opettajalta entistä enemmän suunnittelua, kun sama harjoitus tulisi olla toteutettavissa viidellä eri merkkisellä laitteella.

5.1 Koneeton ohjelmointi koulussa

Ohjelmoinnin opetus voi myös olla koneetonta, toisin kuin usein ajatellaan. Linda Liukas on kirjoittanut kirjan nimeltään Hello Ruby. Kirja on tarkoitettu koodaamisen opetteluun viisivuotiaasta ylöspäin. Kirja sopii sisällöltään ja ominaisuuksiltaan oivallisesti esi- ja alkuopetuksen materiaaliksi ja johdatukseksi ohjelmointiin. (Liukas, 2015) Koodisatukirjan avulla oppilas pääsee tutustumaan ohjelmointiin liittyvään sanastoon ja ohjelmoinnilliseen ajatteluun, mikä onkin esi- ja alkuopetuksen OPS:n mukainen osaamistavoite ohjelmoinnin osalta.

Algoritmit ovat täsmällisiä ohjeita jonkin toiminnon toteuttamiseen. Jotta oppilas oppii ajattelemaan algoritmisesti, hänen on hyvä tutustua muutamaan perusperiaatteita selkeyttävään vaiheeseen koodin luomisesta. Ensiksi täytyy luoda omia algoritmeja tavoitteena saada joku, tai jokin suorittamaan haluttu tehtävä. Toisena harjoituksena on hyvä seurata muiden tekemiä algoritmeja, se herättää kiinnittämään paremmin huomiota ohjeiden täsmällisyyteen. Erityisesti koodikielellä ohjelmoitaessa työskentely on pikkutarkkaa, eikä virheille ole sijaa. Näiden vaiheiden harjoittelu auttaa oppilaita ymmärtämään paremmin koneiden toimintaa. (Snyder, 2006) Vastaavia harjoituksia on helppo toteuttaa esimerkiksi liikuntatunnilla osana koneetonta ohjelmointia, näin ollen virheillekin jää hieman tilaa.

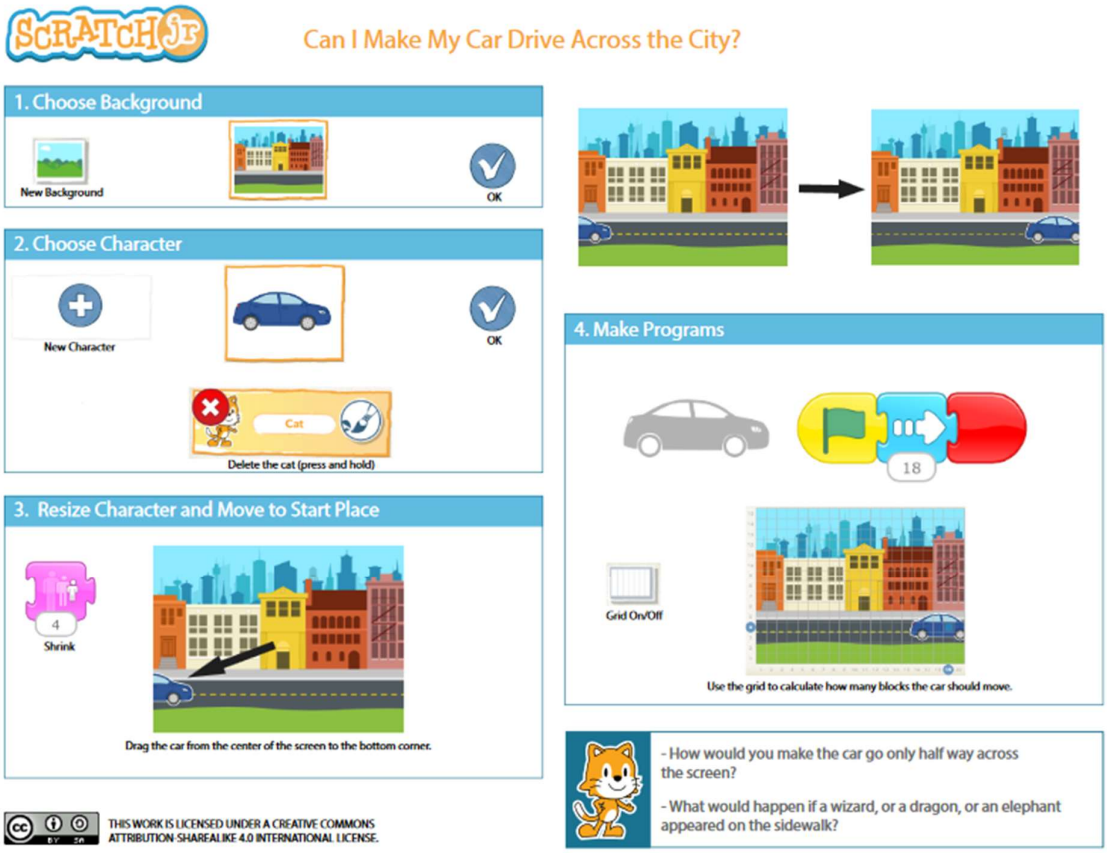
Koneettomaan ohjelmointiin yhdistetään usein ohjelmoinnillisen ajattelun harjoitukset, fyysiset apuvälineet, joiden avulla voidaan tehostaa muistamista, yhteistyö, esimerkkien kontekstualisointi ja scaffolding (Sentance & Csizmadia, 2017). Näillä välineillä pystytään tukemaan koneettoman ohjelmoinnin sisältöjä ja rakentamaan hyvää pohjaa varsinaiselle ohjelmoinnin opettelulle. Yhdistämällä ohjelmointia ja koneettoman ohjelmoinnin harjoituksia, pystytään saavuttamaan progressiivisia oppimiskokemuksia, joissa tieto linkittyy aiempaan ja pystytään syventämään kokonaisvaltaista ymmärrystä ohjelmoinnin saralla.

Samoja vaiheita voidaan harjoitella ohjelmoinnissa niin koneella kuin ilmankin. Hyviä harjoituksia koneettomaan ohjelmointiin kokosivat AlAmer, Al-Doweesh, Al-Khalifa ja Al-Razgan (2015), esimerkiksi funktioiden koneettomaan harjoitteluun leiriläisten (jota varten materiaali

oli koottu), tuli leikata ja kasata paperista hahmo ohjelmoinnillisia ohjeita (esimerkiksi toistolauseita) noudattamalla. Tehtävät suunniteltiin leiriläisten näkökulmasta, tehtävien täytyi herätellä mielenkiintoa aiheeseen ja motivoida itsenäiseen työskentelyyn. Kesäleirin jälkeen lähes kaikki leiriläiset olivat valmiita suosittelemaan leiriä myös ystävilleen (AlAmer;Al-Doweesh;Al-Khalifa;& Al-Razgan, 2015). Kun tunnetaan ohjelmoinnin vaiheet ja ryhmä jolle opetusta suunnitellaan, voidaan sen ohjelmointia integroida koneettomasti vaikka mihin oppiaineeseen. On vain opettajasta kiinni kuinka rohkeita ratkaisuja luokkahuoneessa uskalletaan käyttää.

5.2 Ohjelmoinnin opetusta- ja oppimista tukevat sovellukset ja välineet

Scratch Jr on ilmainen, ohjelmoinnin alkeiden opetteluun kehitetty sovellus. Kouluissa Scratch Jr soveltuu esi- ja alkuopetuksen opetusvälineeksi ja se vaatii opetuskäyttöön soveltuvia tablet-laitteita. Työskentelyä voidaan tehdä pareittain tai ryhmissä, jos laitteita ei ole jokaiselle omaa. Scratch Jr:n avulla oppilaat pääsevät kokeilemaan yksinkertaista koodin tekemistä erilaisten harjoitusten avulla. Kirjoitettava koodi on valmiiksi selkeissä palasissa, joten oppilaiden ei tarvitse osata kirjoittaa sitä itse, vaan koodin kirjoitus tapahtuu valmiita paloja yhdistelemällä. Nämä harjoitukset luovat hyvää pohjaa myös remiksauksen opettelulle, jossa periaatteena on yhdistellä olemassa olevaa koodia uuteen, itse luotuun koodiin, jottei koodin jokaista riviä tarvitse kirjoittaa itse. Scratch Jr:n tehtävissä on helppo vaihdella taitotasoa, alkuun voidaan käyttää tarkkaan ohjeistettuja tehtäviä, joissa korostuvat oppilaiden valintojen kautta esimerkiksi ohjelmointiin liittyvä toistorakenteen harjoittelu (kuva 1) ja edistyneemmät oppilaat voivat hyödyntää projekteissaan useampia hahmoja ja lisätä sinne uusia ominaisuuksia, kuten ääniä tai valintojen (kuten klikkauksen) takana olevia toimintoja. Scratch Jr:iin liittyen on suunniteltu valmiiksi integroitavia projektikokonaisuuksia esimerkiksi matematiikkaan ja äidinkieleen liittyen, joten materiaalit ovat ohjelmointia aloittelevalle opettajallekin helppo tapa aloittaa ohjelmoinnin opetuskäyttö. (Scratch Jr, 2017)



Kuva 1. (Scratch Jr, 2017)

Scratch on suosittu visuaalinen ohjelmointiympäristö, joka on kehitetty MIT:ssä ja on ilmaiseksi kaikkien käytettävissä ja sen sisällöt ovat helposti jaettavissa. Scratchin avulla koodaamisen opettelu on huomattavasti helpompaa kuin ohjelmointikielten (kuten C-kielen) käyttäminen, kun ohjelmointia tehdään isoilla paloilla pikkutarkan koodin sijaan. Se on myös suomennettu jo suurelta osin. Scratchin sisältöjä voi hyödyntää opetuksessa minkä ikäisten oppilaiden kanssa tahansa, mutta tärkeää on tuntea oppilaiden lähtötasot ohjelmointiin. Työstämisen voi aloittaa vanhempien oppilaidenkin kanssa Scratch Jr:llä, mutta kun perustaidot on opittu, tarjoaa Scratch hieman enemmän haastetta. Scratchissä tulee mukaan myös yhteisöjä, jotka voivat toimia joillekin oppilaille hyvänä motivaattorina. Perusopetuksen puolella hyvä siirtymävaihe Scratchiin on viimeistään kolmosluokalla. Scrath on selainpohjainen, joten laitteiston näkökulmasta joissain kouluissa paremmin saatavilla kuin tablet-sovellukset. Opettaja voi luoda Scratchiin omat tunnuksensa ja luoda sieltä tarvitsemansa ryhmät projekteja varten. Scratch toimii opettajan näkökulmasta siinäkin mielessä paremmin, että opettajan on helppo tarkastella oppilaiden töitä ja seurata edistymistä. Opettaja pystyy myös tarvittaessa muokkaamaan ja poistamaan oppilaiden sisältöjä, jos ne sisältävät sopimattomia asioita. Samoin kuin Scratch Jr:ssä,

Scratchissä on valmiita projekteja, joista on helppo aloittaa työskentely taitotason mukaisesti. Tehtäviin saa lisättyä haastetta uusilla ominaisuuksilla. Scratchissä luodut sisällöt muistuttavat entistä enemmän ohjelmointikielten rakennetta ja harjoituksissa tulee opittua lähes huomaamatta esimerkiksi ohjelmoinnin silmukoita ja toistorakenteita ja debuggausta, eli virheiden paikantamisen sisältöjä. (Scratch, 2017)

Code.org on vuonna 2013 alkunsa saanut sivusto, jonka tavoitteena on tuoda TVT:n opetus helpommin lähestyttäväksi kaikille, niin opettajille, kuin oppilaille, ympäri maailman. Code.org tarjoaa alustan, jossa voi rakentaa oppilaille projekteja graafisessa ohjelmointiympäristössä. Sisältöjä on Code.org:ssa esi- ja alkuopetuksesta aina lukioon saakka. Materiaalia löytyy niin koneettomasta ohjelmoinnista kuin nopeutetuista tietojenkäsittelytieteiden sisällöistäkin ja Code.org on suomennettu hyvin pitkälle, työtä hankaloittaa se että uutta opetusmateriaalia tulee koko ajan lisää. Oppilaita motivoivat tutut hahmot, kuten Angry Birds. Kuten Scratchissä, Code.org:ssa koodia rakennetaan valmiista palasista. Hyvä lisäominaisuus on mahdollisuus tarkastella palikoista rakennettua koodia myös tekstinä, eli versiona joka muistuttaa hyvin paljon ohjelmointikielten rakennetta. Code.org:n (kuten myös Scratchin) sisällöt sopivat hyvin peruskoulun kuudennen luokan tavoitteiden saavuttamiseen. (Code.org, 2017)

Roboteista esimerkiksi Blue-Bot on oivallinen väline esi- ja alkuopetukseen. Robottia voidaan ohjata joko bluetoothin kautta tabletilla tai selaimella tai manuaalisesti sen päällä olevilla painikkeilla. Tavoitteena on saada robotti kulkemaan haluttu reitti joko valmiilla matolla, tai itse vaikka maalarinteipistä tehdyllä radalla. Blue-Botin avulla oppilaat pääsevät tutustumaan esimerkiksi koodin kirjoittamiseen ja ongelmanratkaisuun debuggauksen kautta. Blue-Botin avulla ohjelmointia on helppo integroida muihin oppiaineisiin, kun robotti laitetaan kulkemaan reittiä esimerkiksi tavu-vihjeiden perusteella. (Bee-Bot, 2017)

Ohjelmoinnin opettamiseen peruskouluissa on kehitetty lukuisia sovelluksia, materiaaleja ja välineitä ja tässä kappaleessa esitellyt ovat vain muutamia pintaraapaistuja esimerkkejä niistä. Kun oppitunnin tavoitteet on mietitty tarkkaan, voidaan etsiä sopivat välineet niiden saavuttamisen tueksi. Aiemmin mainittu integroitavien sisältöjen ja välineiden tunteminen on tärkeää, jotta oppitunnista saadaan toimiva ja pedagogisesti merkityksellinen kokonaisuus, eikä välineillä päädytä aina vain leikkimään ilman suurempaa tavoitehakuisuutta.

6 JOHTOPÄÄTÖKSET

Tieto- ja viestintäteknologia on pitkään ollut osa suomalaista opetussuunnitelmaa ja uuden opetussuunnitelman (2014) myötä, ohjelmoinnin opetus tuli osaksi sitä. TVT:n opetuskäytössä olisi hyvä pyrkiä pois vanhasta mallista, jossa tietotekniikkaa hyödynnetään vain mekaanisesti ja ohjata sitä interaktiivisempaan suuntaan, jossa tvt:n avulla tehdään muutakin kuin kirjoitetaan puhtaaksi aiempia tuotoksia. Ohjelmoinnin oppiainesisältöjen avulla opetusta voidaan toteuttaa niin pareittain kuin yhteisöllisestikin. Ohjelmoinnin opetuksessa keskitytään algoritmiseen ajatteluun ja ongelmanratkaisutaitoihin, taitoihin joita pidetään keskeisinä määrittelyinä ohjelmoinnilliselle ajattelulle.

”Kognitiiviset perustaidot, kuten lukeminen, laskeminen ja kirjoittaminen, eivät enää yksin riitä muuttuvan yhteiskunnan haasteiden kohtaamisessa, vaan yksilöltä vaaditaan yhä enemmän oppimaan oppimisen taitoja – yksilön tulee osata oppia ja ajatella itsenäisesti” (Järvelä;Häkkinen;& Lehtinen, 2006, s. 40). Ohjelmoinnillisella ajattelulla eli oppilaan ongelmanratkaisutaitoja kehittämällä voidaan tukea oppilaan kognitiivisten taitojen kehitystä. Oppilaat oppivat pilkkomaan ongelmia osiin ja ratkaisemaan niitä vaihe kerrallaan. Tämän mallin avulla oppilaat ovat kyenneet ratkaisemaan jopa omaa taitotasoaan vaativampia tehtäviä. Mallia on osattu myöhemmin soveltaa muihin oppiaineisiin, mikä osoittaa, että ohjelmoinnillisesta ajattelusta on hyötyä yli oppiainerajojen. Tarjoamalla laadukasta ohjelmoinnin opetusta, voidaan siis saada hyviä tuloksia aikaiseksi myös muissa oppiaineissa.

Ohjelmoinnillisen ajattelun opettamiseen liittyviä käytännön haasteita ja tutkimuksen puutetta on vielä paljon, sillä osaavia opettajia ei ole vielä tarpeeksi ja opetustavat ja -vaiheet vaativat hiomista. Matematiikan opetuksesta on vuosien saatossa saatu oppiaine, jossa opetus etenee progressiivisesti, aiemmin opittu sisältö tukee uuden oppimista, ohjelmoinnin opetukseen pitäisi saada vastaava malli. (Wing, 2016) Mitä pidempään ohjelmointi on mukana opetussuunnitelmassa, sitä paremmaksi oppiainesisällötkin hioutuvat. Käytännön kautta pystytään parhaiten näkemään, missä iässä on parasta aloittaa graafisten ohjelmointiympäristöjen käyttäminen ja mitkä sisällöt opitaan parhaiten koneettomasti ja mitkä taas jossain ohjelmointiympäristössä. Monipuoliset ja oppilaiden mielenkiinnon mukaan suunnitellut sisällöt sitouttavat ja innostavat oppilaita tehtäviin.

Koodi2016-oppaassa todetaan, että niille jotka innostuvat ohjelmoinnista, on annettava mahdollisuus kehittyä eteenpäin (Liukas & Mykkänen, 2014). Opettajat pääsääntöisesti toivovat

pystyvänsä rohkaisemaan oppilaitaan heidän vahvuuksissaan. Usein ensimmäisenä haasteena tulee vastaan aika, sillä tuntisuunnitelmien teossa pitäisi pystyä ottamaan huomioon koko luokkahuoneen kirjo. Varsinkin isoissa kouluissa jo pelkkä peruspaketin tuottaminen voi aiheuttaa todella paljon työtä oppilaiden monikulttuurisuuden, mahdollisen dyskalkulatian ja ihan vain erilaisten elämäntilanteiden vuoksi. Koulun resurssit saattavat tulla vastaan myös rahan puolesta, aina ei ole mahdollista järjestää lisää tarjontaa edes valinnaisina kursseina. Perusopetuksen valinnaisten opintojen yhteisenä tehtävänä on syventää oppimista, laajentaa opintoja ja vahvistaa jatko-opintovalmiuksia (Opetushallitus, 2015). Valinnaisten tuntien määrä vaihtelee luokka-asteittain ja päätöksen niiden käytöstä tekee opetuksen järjestäjä. Valinnaisilla kursseilla on hyvä syventää osaamistaan. Nähtäväksi jää, missä kouluissa ohjelmointia tarjotaan valinnaisena aineena. Haasteena on varmistaa, etteivät oppilaat, jotka opiskelevat ohjelmointia valinnaisena, kyllästy perustunneilla. Tuntisuunnitelmiin pitäisi saada selkeä ero, eivätkä sisällöt voi olla päällekkäisiä. Tarvittaessa edistyneemmät oppilaat voivat toteuttaa pidemmälle edenneitä töitään muilla tunneilla, etenkin, jos opettaja on sama. Mielenkiinnolla odotan lisääkö ohjelmoinnin opetus peruskouluissa sen suosiota valinnaisena aineena lukion puolella.

Ohjelmointia opetettaessa on tärkeää muistaa, ettei se ole vain koodikielen kirjoittamista, vaan ohjelmointia voidaan harjoitella myös koneettomasti. Alla olevassa taulukossa on vertailtu eroja ja samankaltaisuuksia luvuissa 5 ja 6 käsiteltyjen ohjelmoinnin ja koneettoman ohjelmoinnin välillä 5-6 luokkalaisten oppiainesisältöihin soveltaen.

	Vaiheet	**Debuggaus**	**Remiksaus**
Ohjelmointi	Suunnittelu, kirjoittaminen, kääntäminen, debuggaus.	Pikkutarkkaa, virheille ei ole tilaa. Tarkin koodin virheiden läpikäynti ohjelmaa käännettäessä.	Hyödynnetään olemassa olevaa koodia ja jatketaan mahdollisesti omaa siihen päälle. Esimerkiksi Scratch-ympäristössä valitaan ohjelmalle jonkun muun tekemä pohja ja siihen koodataan oma peli jatkoksi.
Koneeton ohjelmointi	Vaiheet samat kuin ohjelmoinnissa, harjoituksia on kuitenkin helpompi pilkkoa osiin ja	Paljon sallivampi virheille koodissa. Konkreettisia harjoituksia voidaan tehdä	Harjoituksia voidaan toteuttaa vaikka pariohjelmointina, toinen aloittaa koodin, toinen tekee sen loppuun,

	harjoitella vaikka vain funktioiden toimintaa.	esimerkiksi liikuntatunnilla toimintaohjeiden avulla.	valmiita osia voidaan ottaa myös muilta ryhmiltä.
Samankaltaisuudet	Toimivan koodin saavuttamiseksi tulee vaiheita pyörittää ympyrässä kunnes ohjelma toimii. Koneettomalla ohjelmoinnilla voidaan harjoittelu eriyttää helpommin koskemaan vaikka vain toistorakenteita.	Perusperiaatteena oppia hahmottamaan tarkan koodin merkitys ohjelmalle, pienikin virhe saattaa johtaa suurempiin ongelmiin.	Tärkein ajatus se, ettei pyörää tarvitse keksiä uudelleen. Jos joku on luonut jo toimivan hyppyohjeen, voidaan käyttää valmista pohjaa eikä jokaista koodiriviä tarvitse kirjoittaa itse.

Taulukko 1.

Yhteenvetona voidaan todeta ohjelmoinnin olevan tulevaisuudessa tärkeä oppiaine, jonka sisältöjä jokaisen olisi hyvä osata. Ohjelmoinnillinen ajattelu harjoittaa tulevaisuuden ajattelu- ja ongelmanratkaisutaitoja, kun ongelmaa pyritään pilkkomaan osiin ratkaisun saavuttamiseksi. Sen avulla voidaan saavuttaa parempia oppimistuloksia useissa oppiaineissa, eikä pelkästään ohjelmoinnin ja matematiikan saralla. Suomalaisissa peruskouluissa ohjelmoinnista ei ole tehty omaa oppiainetta, vaan sen toteutus järjestetään integroinnin avulla, tämä on varsinkin alakoulun puolella tärkeää, sillä ohjelmoinnille on varattu vain muutamia vuosiviikkotunteja opetussuunnitelmassa. Pääpaino ohjelmoinnissa on matematiikan oppiainesisällöissä, mutta juuri integroinnin avulla sisältöjä saadaan yhdistettyä muihinkin oppiaineisiin, kuten käsitöihin ja liikuntaan. Ohjelmoinnin opettelu harjoittaa myös oppilaiden monilukutaitoa, kun oppilaat pääsevät tulkitsemaan täysin uudenlaisia tekstejä ja kokonaisuuksia.

Kuten aiemmin mainittua, ohjelmoinnin ja ohjelmoinnillisen ajattelun parista puuttuu vielä paljon tutkimusta. Mielenkiintoista olisi tutkia, millaisia tuloksia ohjelmoinnin opettamisella on lyhyellä aikavälillä saavutettu. Mahdollisena pro gradu-tutkielman aiheena pitäisin myös pariohjelmoinnin mahdollisuuksia. Kandidaatin työssä toteuttamassani kirjallisuuskatsauksessa löysin muutamia tutkimustuloksia, joiden mukaan pariohjelmoinnin hyödyt olivat verrattavissa ohjelmoinnillisen ajattelun hyötyihin ja vahvistivat niitä, oppilaat kokivat tehtävät hauskem-

pina, he sitoutuivat tehtäviin paremmin ja saavuttivat parempia tuloksia vertaistuen avulla. Haluaisin tutkia, onko mallia kokeiltu jo Suomessa ja havaittiinko sillä samoja vaikutuksia myös täällä.

7 LÄHDELUETTELO

AlAmer, R.;Al-Doweesh, W.;Al-Khalifa, H.;& Al-Razgan, M. (2015). Programming Unplugged: Bridging CS Unplugged Activities Gap for Learning Key Programming Concepts. *Fifth International Conference on e-Learning* (ss. 97-103). Riyadh: College of Computer and Information Sciences, King Saud University.

Alessi, S.;& Trollip, S. (1985). *Computer-based instruction.* New Jersey: Prentice-Hall, Inc.

Clements, D.;& Gull, D. (1984). Effects of Computer Programming on Young Children's Cognition. *Journal of Educational Psychology* , 1051-1058.

Denning, P. (2017). Remaining Trouble Spots with Computational Thinking. *Communications of the ACM*, 33-39.

Forret, M.;Harlow, A.;& Taylor, M. (2010). Using a Computer Programming Environment and an Interactive Whiteboard to Investigate Some Mathematical Thinking. *Procedia Social and Behavioral Sciences*, 561-570.

Google, Microsoft, Facebook ja Twitter. (8. Marraskuu 2017). *Code.org*. Noudettu osoitteesta https://code.org/

Haapasalo, L. (1994). *Oppiminen, tieto & ongelmanratkaisu*. Jyväskylä: Gummerrus Kirjapaino Oy.

Haasio, A.;& Haasio, M. (2008). *Pulpetit virtuaalivirrassa.* Jyväskylä: Gummerrus Kirjapaino Oy.

Järvelä, S.;Häkkinen, P.;& Lehtinen, E. (2006). *Oppimisen teoria ja teknologian opetuskäyttö.* Helsinki: WSOY Oppimaateriaalit Oy.

Kupiainen, R.;Kulju, P.;& Mäkinen, M. (2015). Mikä monilukutaito? Teoksessa T. Kaartinen, *Monilukutaito kaikki kaikessa* (ss. 13-24). Tampere: Tampereen yliopistopaino - Juvenes Print.

Lehtinen, T. (26. Lokakuu 2015). Opettajat opettelevat nyt koodaamaan. *Helsingin Sanomat*.

Leppäaho, H. (2007). *Matemaattisen ongelmanratkaisutaidon opettaminen peruskoulussa.* Jyväskylä: Jyväskylä University Printing House.

Lifelong Kindergarden Group, M. (8. Marraskuu 2017). *Scratch*. Noudettu osoitteesta https://scratch.mit.edu/

Liukas, L. (2015). *Hello Ruby.* Keuruu: Otavan Kirjapaino Oy.

Liukas, L.;& Mykkänen, J. (2014). *Koodi2016 Ensiapua ohjelmoinnin opettamiseen peruskouluissa.* Helsinki: Lönnberg Print.

Opetushallitus. (2015). *Perusopetuksen opetussuunnitelman perusteet.* Tampere: Juvenes Print - Suomen yliopistopaino Oy.

Opetushallitus. (22. Toukokuu 2016). *OPS2016 työryhmät*. Noudettu osoitteesta OPS 2016 - Esi- ja perusopetuksen opetussuunnitelman perusteiden uudistaminen: http://www.oph.fi/ops2016/tyoryhmat

Rheingold, H. (2003). *Mobiilijoukot.* Jyväskylä: Gummerrus Kirjapaino Oy.

Saez-Lopez, J.-M.;Roman-Gonzales, M.;& Vazquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scrach" in five schools. *Computers & Education*, 129-141.

Sanger, J.;Willson, J.;Davies, B.;& Whittaker, R. (1997). *Young children, videos and computer games.* Lontoo: Biddles Ltd.

Sentance, S.;& Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Educ Inf Technol* , 469-495.

Snyder, L. (2006). *Fluency with information technology: skills, concepts & capabilities.* Boston: Pearson Education, Inc.

Tella, S.;Vahtivuori, S.;Vuorento, A.;Wager, P.;& Oksanen, U. (2001). *Verkko opetuksessa – opettaja verkossa.* Helsinki: Edita Oyj.

Terrapin Software. (8. Marraskuu 2017). *Bee-Bot*. Noudettu osoitteesta https://www.bee-bot.us/

the DevTech Research Group at Tufts University; the Lifelong Kindergarten Group at the MITMediaLab; the Playful Invention Company. (8. Marraskuu 2017). *Scratch Jr.* Noudettu osoitteesta https://www.scratchjr.org/activities/card01-car.pdf

the DevTech Research Group at Tufts University; the Lifelong Kindergarten Group at the MITMediaLab; the Playful Invention Company. (8. Marraskuu 2017). *Scratch Jr*. Noudettu osoitteesta https://www.scratchjr.org/

Tilastokeskus. (22. Toukokuu 2016). *Tilastokeskuksen tilasto Suomen kouluista jaoteltuna koon mukaan.* Noudettu osoitteesta Tiedot perustuvat Tilastokeskuksen oppilaitosrekisteriin: http://pxnet2.stat.fi/PXWeb/pxweb/fi/S

Wing, J. (Maaliskuu 2006). Computational thinking. *Communications of the ACM*, 33-35.

Wing, J. (23. 3 2016). *Computational thinking, 10 years later*. Noudettu osoitteesta Microsoft Research Blog: https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/