Ohjelmoinnin opetus valtakunnallisella MAA11 opintojaksolla

Tapio Nojonen

Pro gradu -tutkielma
Syyskuu 2024

Tarkastajat:
Prof. Vesa Halava
FT Anni Hakanen

MATEMATIIKAN JA TILASTOTIETEEN LAITOS

Turun yliopiston laatujärjestelmän mukaisesti tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck-järjestelmällä

TURUN YLIOPISTO
Matematiikan ja tilastotieteen laitos

Tapio Nojonen: Ohjelmoinnin opetus valtakunnallisella MAA11 opintojaksolla
Pro gradu -tutkielma, 31 s., 2 liites.
Matematiikka
Syyskuu 2024

Tässä tutkielmassa tutkitaan ohjelmoinnin opetusta lukion pitkän matematiikan opintojaksolla MAA11, joka on ainoa vuoden 2019 lukion opetussuunnitelman perusteissa mainittu ohjelmointia sisältävä opintojakso. Tutkielman päätavoite on selkeyttää sitä, miten ohjelmoinnin opetusta toteutetaan kyseisellä opintojaksolla, sillä pelkän opetussuunnitelmatekstin perusteella ei synny selkeää kuvaa siitä, opetetaanko ohjelmointia lainkaan saati millä tavoin ja mitä teknologioita hyödyntäen. Tämän takia mielenkiinnon kohteena on erityisesti ohjelmointikielten valinta, opetettavien ohjelmointiteknologioiden valinta sekä opettajien suhtautuminen ohjelmointiin, ja onko tällä vaikutusta siihen, miten ohjelmointia opetetaan.

Tutkielma on toteutettu sähköisenä kyselytutkimuksena, joka on vuoden 2024 toukokuussa lähetetty sähköpostitse lähes kaikkiin Suomen lukioihin. Itse kysely on luotu RedCAP-ohjelmistolla, josta kerättyä dataa on analysoitu taulukkolaskentasovelluksilla. Eri vastausvaihtoehtojen välillä on etsitty vaikuttavuuksia ja lisäksi kaikkien muuttujien välillä on laskettu Pearsonin korrelaatiokertoimet.

Kyselyyn vastasi yhteensä 74 opettajaa 61 eri lukiosta, mikä kattaa hieman alle viidenneksen kaikista Suomen lukioista. Vastausten perusteella kaikissa lukioissa opetetaan ohjelmointia Pythonilla ja kaikissa kyselyyn vastanneissa lukioissa oppilaille opetetaan sekä ehto- että toistolauseiden käyttöä. Valtaosa opettajista näkee ohjelmoinnin omana kokonaisuutenaan, joka on yhtä tärkeä kuin MAA11 kurssin matemaattinen sisältö. Lisäksi yli puolessa kaikista kyselyyn vastanneista lukioista ohjelmointia opetetaan opintojaksolla vähintään seitsemän 75 minuutin oppitunnin verran. Melkein kaikkien muuttujien väliset korrelaatiokertoimet ovat joko mitättömiä tai pieniä. Jos opettaja näkee ohjelmoinnin vähemmän tärkeänä kokonaisuutena kuin MAA11 kurssin matemaattisen sisällön, on tällä keskivahva negatiiviinen korrelaatio sen kanssa, että oppilaat ohjelmoivat itsenäisesti opintojakson aikana. Lisäksi oppimateriaalien ja käytettyjen opetusmetodien välillä on pieniä korrelaatioita.

Tutkielman tulosten perusteella jatkotutkimuksessa olisi hyvä kiinnittää tarkempaa huomiota ohjelmoinnin opetusta sisältävien oppituntien määrään ja siihen, miten määrä vaikuttaa esimerkiksi valittuihin oppimateriaaleihin. Samalla perusteltua olisi tutkia laajemmin opettajien suhtautumista ohjelmointiin ja miten suhtautuminen vaikuttaa matematiikan opetukseen integroituun ohjelmoinnin opetukseen.

Asiasanat: lukio, ohjelmointi, matematiikka, opetus, opetussuunnitelma

Sisällys

1 Johdanto **1**

2 Ohjelmoinnin opetus **2**
2.1 Ohjelmoinnin opetuksen metodiikka 3
2.2 Ohjelmoinnin opetus matematiikassa 5
2.3 Ohjelmoinnin opetus Suomessa 9

3 Tutkimuskysymykset, tutkimuksen tarkoitus ja tutkimusmenetelmät **12**
3.1 Tutkimusaineisto 12
3.2 Analyysi 14

4 Vastausten analyysi **15**
4.1 Oppituntimäärän vaikutus muihin muuttujiin 20
4.2 Muita muuttujien välisiä havaintoja 25

5 Yhteenveto **29**
5.1 Pohdinta ja ehdotuksia jatkotutkimukselle 30

Lähdeluettelo **32**

Liitteet **38**

A Kyselylomake **38**

1 Johdanto

2020-luvulla on mahdotonta välttää tietotekniikkaa. Tietokoneet ja erilaiset älylaitteet ovat tulleet osaksi lähes jokaisen ihmisen elämää niin töissä kuin vapaa-ajalla. Täten ei ole yllätys, että tietotekniikan merkitys koulutuksessa on korostunut valtavasti 2000-luvun aikana. Luonnollisesti tämä tarkoittaa esimerkiksi paperilla kirjoittamisen korvaamista tekstinkäsittelyohjelmalla ja oppimateriaalien jakamista erilaisilla internetissä toimivilla alustoilla.

Ei kuitenkaan riitä, että osaa ainoastaan käyttää erilaisia sovelluksia tai kirjoittaa tekstiä näppäimistöllä. Jo pitkään on pidetty tärkeänä, että oppilaat saavat koulusta sellaisia taitoja, jotka palvelevat heitä hyvin tulevaisuudessa ja erityisesti työelämässä. Näitä taitoja on painotettu eri aikoina eri tavoin, mutta nyt 2000-luvulla ovat korostuneet osuvasti nimetyt 2000-luvun taidot, joihin kuuluu esimerkiksi ongelmanratkaisu ja looginen ajattelu (Balanskat ja Engelhardt 2015, s. 15). Yhdeksi parhoista tavoista hioa näitä taitoja on osoittautunut ohjelmoinnillisen ajattelun opetus ja tästä syystä ohjelmoinnin opetuksen merkitys on kasvanut valtavasti viime vuosien aikana (Balanskat ja Engelhardt 2015; Bocconi et al. 2022).

Ei ole olemassa yhtä oikeaa tapaa opettaa ohjelmointia ja tästä syystä on olemassa lukuisia eri tapoja sisällyttää ohjelmointi opetussuunnitelmiin. Suomessa on tehty päätös integroida ohjelmoinnin opetus muihin oppiaineisiin sekä peruskoulussa että lukiossa. Pääsääntöisesti ohjelmointi on päätetty sijoittaa matematiikan oppiaineeseen. Ohjelmointi eroaa perinteisesti matematiikassa opetettavista aiheista ja esimerkiksi valtaosa lukioista sisällyttää ohjelmointia omaan, matematiikasta irralliseen, oppiaineeseen, kuten tietotekniikkaan (Nojonen 2024). Realiteetti kuitenkin on, että lukiotasolla ohjelmointi on mainittu ainoastaan yhdessä pitkän matematiikan opintojaksossa. Tämän tutkielman tavoitteena onkin raottaa, miten ohjelmoinnin opetus on integroitu tähän matematiikan opintojaksoon ja miten tämä opetus toteutetaan.

Tutkielman rakenne on seuraavanlainen: luvussa 2 esitellään taustaviitekehys, joka käy läpi ohjelmoinnin opetuksen metodiikkaa, miten ohjelmoinnin opetus voidaan integroida matematiikan opetukseen ja mitä Suomen viralliset opetussuunnitelmat sanovat ohjelmoinnin opetuksesta. Luvussa 3 esitellään tutkimuksen tutkimuskysymykset, -menetelmät, -aineisto ja miten aineistonkeruu sekä aineiston analyysi on toteutettu. Luvussa 4 esitellään analyysin tulokset ja luvun 5 yhteenvedossa pohditaan saatuja tuloksia ja ehdotetaan jatkotutkimuksen kohteita.

2 Ohjelmoinnin opetus

Ohjelmoinnin opetusta on järjestetty yksittäisissä kouluissa aina 1960-luvulta lähtien, kun tietokoneet olivat vielä varsin harvinaisia kotikäytössä (Forneck 1990, Hubwieser et al. 2011, s. 29). Joissain valtioissa, kuten Israelissa, ohjelmoinnin opetuksen edut nähtiin varsin varhaisessa vaiheessa ja ohjelmointia sisällytettiin kansallisiin opetussuunnitelmiin jo 1970-luvulla (Balanskat ja Engelhardt 2015, s. 37). Ohjelmoinnin opetus ei kuitenkaan ole vakiintunut valtaosaan maista jo 1970-luvulla, vaan asiaan on herätty vasta paljon myöhemmin. Esimerkiksi Englannissa tietotekniikkaa ja ohjelmointia pidettiin kyllä tärkeinä vuosituhannen vaihteessa ja ne tulivat osaksi kansallista opetussuunnitelmaa, mutta jo 2010-luvun alussa opetuksen taso oli laskenut huomattavasti ja kouluissa opetettiin ainoastaan tietoteknisten laitteiden käyttöä, eikä varsinaisesti ohjelmointia tai tietotekniikkaa (Furber 2012, s. 4-8). (Nojonen 2024.)

Maailma on kuitenkin digitalisoitunut viimeisten vuosikymmenten aikana niin hurjaa vauhtia, että monissa maissa on nähty tarpeen opettaa tietotekniikkaa ja ohjelmointia kouluissa. Erityisesti halutaan lisätä nuorten kykyä pärjätä tulevaisuuden työmarkkinoilla, mikä on innostanut maailmanlaajuisesti sekä lisäämään teknologian käyttöä opetuksessa että antamaan tiedot ja taidot ymmärtää, miten nämä teknologiat toimivat. (Balanskat ja Engelhardt 2015; Bocconi et al. 2022.) Ohjelmoinnin merkityksen kasvuun on vaikuttanut myös se, miten ohjelmoinnista puhutaan ja mitä hyötyjä sen osaamisella näetään olevan. Seymour Papert esitti kirjassaan (Papert 1980) alun perin konseptin 'rattaista', joilla hän kuvainnollisti tietokoneohjelmoinnissa käytettävää ajattelutapaa, jota voi hyödyntää muussakin ongelmanratkaisussa. Myöhemmin Jeanette Wing artikkelissaan (Wing 2006) esitti hyvin samankaltaisiin ideoihin pohjautuvan termin 'ohjelmoinnillinen ajattelu'. Termi ei ole synonyymi ohjelmoinnille, vaan tarkoittaa ohjelmoinnille tyypillisiä ajattelun taitoja, joita voidaan hyödyntää myös monissa muissa konteksteissa (Wing 2006). Sekä Papertin että Wingin ideat ovat nousseet myöhemmin valtavirtaan ja erityisesti ohjelmoinnillinen ajattelu on käsitteenä levinnyt laajalle. Ohjelmoinnillisen ajattelun taitoja pidetään hyvin tärkeinä tulevaisuudessa pärjäämisen kannalta, mikä on syy sille, että ohjelmoinnin opetuksen lisääminen opetussuunnitelmiin on nähty hyödyllisenä (Balanskat ja Engelhardt 2015).

Ohjelmoinnin opetuksen lisääminen opetussuunnitelmiin ei kuitenkaan ole toteutunut kaikkialla samalla tavalla ja eri koulutusjärjestelmien välillä onkin suuria eroja siinä, mitä opetuksessa painotetaan. Esimerkiksi suuria eroja on siinä, opetetaanko ohjelmointia laisinkaan vai keskitytäänkö opetuksessa pelkästään ohjelmoinnillisen ajattelun taitoihin tai onko omaa oppiainetta, joissa näitä nykymaailmalle

tärkeitä taitoja opetetaan. (Bocconi et al. 2022). Niinpä tarkastaltaessa ohjelmoinnin opetusta Suomessa on tärkeää ensin määrittää, mitä ohjelmoinnin opetuksella itseasiassa tarkoitetaan ja miten juuri ohjelmointia voidaan opettaa matematiikan oppiaineessa.

2.1 Ohjelmoinnin opetuksen metodiikka

Poiketen monesta muusta luonnontieteestä, ohjelmoinnin opetukseen tai ylipäätään tietojenkäsittelyyn ei ole kehittynyt yleisesti vallalla olevaa didaktiikkaa (Belmar 2022, Nojonen 2024, s. 7). Täten ohjelmoinnin opetus saattaa erota merkittävästi paitsi maiden välillä, niin myös yksittäisten koulujen välillä. Tämä ei kuitenkaan tarkoita, etteikö eri puolella maailmaa olisi yritetty luoda standardeja ohjelmoinnin opetukselle tai etteikö ohjelmoinnin opetuksessa eri paikoissa olisi havaittavissa mitään yhteneväisyyksiä. Esimerkiksi Yhdysvalloissa vaikuttava Computer Science Teachers' Association (CSTA) on luonut suosituksia Pohjois-Amerikkalaiseen opetussuunnitelmakontekstiin ja edistänyt ohjelmoinnin opetuksen sisällyttämistä eri koulutusasteille (*CSTA* 2024, Nojonen 2024, s. 7). Nämä suositukset ja standardit ovat myös konkreettisesti vaikuttaneet opetussuunnitelmiin, kuten voidaan nähdä esimerkiksi Fislerin ja muiden vuonna 2021 julkaistusta artikkelista. Ohjelmoinnin opetuksen standardeja on myös esitelty esimerkiksi Dagiénen ja muiden vuoden 2021 artikkelissa, jossa on luotu opetussuunnitelmamalli tietojenkäsittelyn opetukseen eri koulutusasteilla.

Näissä erilaisissa standardeissa on pilkottu ohjelmoinnin osia erilleen toisistaan ja tämän jälkeen kategorisoitu ne sen perusteella, minkä ikäisille oppijoille kyseisen osan opettaminen olisi mielekästä. CSTA:n standardeissa on kategoriat viidelle eri ikäryhmälle ja standardeja yhteensä 120 (Seehorn et al. 2017), kun taas Dagiénen ja muiden opetussuunnitelmamallissa ohjelmoinnille löytyy vain 13 standardia, mutta ikäsuosituksille on ilmoitettu ainoastaan alarajat (Dagiene, Hromkovic ja Lacher 2021, s. 350-351). Esimerkiksi CSTA:n standardeissa suositellaan silmukoita ja ehtolauseita käyttävien ohjelmien luomista 8–11-vuotiaille ja puolestaan Dagiénen ja muiden mallissa silmukoiden käyttöä ilman muuttujia harjoiteltaisiin jo yli seitsenvuotiaiden kanssa, mutta ehtolauseita vasta yli 13-vuotiaiden kanssa.

Nämä esitetyt standardit eivät kuitenkaan ole kaikki kaikessa. Molemmat on luotu käytettäviksi läpi koulutusjärjestelmän ja täten niiden suora hyödyntäminen lukiokontekstissa on vaikeaa. Esimerkiksi CSTA:n standardeissa ehdotetaan algoritmien tehokkuuden ja selkeyden tarkastelemista lukiotasolla, mutta tämä ei välttämättä ole mielekästä, jos oppilailla ei ole tätä varten riittäviä pohjataitoja aikaisemmilta kouluasteilta. Täten ei ole kovinkaan mielekästä tarkastella MAA11 opin-

tojaksoa ja olettaa, että ohjelmoinnin opetus painottuisi juuri CSTA:n standardien mukaisesti, koska kyseessä on lukiotason kurssi. Esitettyjä standardeja voidaan kuitenkin hyödyntää siinä, että ne kertovat melko kattavasti oleelliset ohjelmoinnin taidot, joita oppilaille tulisi opettaa ja luokittelevat aiheita jonkinlaiseen opetettavaan järjestykseen.

Ohjelmoinnin opetuksessa hyvin tärkeitä ovat opetuksessa käytetyt teknologiat. Esimerkiksi, jos opetuksessa käytetään yksinomaan jotakin palikkapohjaista ohjelmointikieltä, kuten Scratchia, niin kyseessä on todennäköisesti ohjelmoinnin alkeiskurssi. Palikkapohjaisessa ohjelmoinnissa oppilaat eivät joudu kamppailemaan minkään yksittäisen ohjelmointikielen syntaksin kanssa, sillä ohjelmat rakennetaan valmiista visuaalisista palikoista, joita yhdistelemällä saadaan aikaan suoritettava tietokoneohjelma. Täten oppilaat voivat keskittyä yksinomaan ohjelmoinnin opetteluun sekä ohjelmoinnillisen ajattelun taitojen kehittämiseen. (Nojonen 2024, s. 9.)

Tekstipohjaisia ohjelmointikieliä voidaan käyttää käytännössä missä tahansa ohjelmoinnin opetuksen vaiheessa, mutta yleensä tekstipohjaisia ohjelmointikieliä suositaan alkeiskurssien jälkeen ja silloin, jos ollaan kiinnostuneita nimenomaan ohjelmoinnin opetuksesta (Weintrop ja Wilensky 2019). Yleisimmät ohjelmoinnin opetuksessa käytetyt tekstipohjaiset ohjelmointikielet ovat Python ja Java (Szabo et al. 2019). Näiden yleisesti käytettyjen ohjelmointikielten välillä on myös selviä eroja ohjelmoinnin opetuksessa. Esimerkiksi on havaittu, että oppimisen kannalta suotuisampaa on ensin käyttää Pythonia ja vasta tämän jälkeen Javaa, sillä tämä on ainakin aloitteleville ohjelmoijille helpompaa (Mannila, Peltomäki ja Salakoski 2006). Käytännön tasolla on myös näyttöä, että siirtyminen Pythonista Javaan on toimiva ratkaisu (Kaila et al. 2018). Myös opetettavilla ohjelmointitekniikoilla on paljon merkitystä ja on esimerkiksi paljon yleisempää ja tehokkaampaa opettaa ensin proseduraalista ohjelmointia ja vasta tämän jälkeen siirtyä olio-ohjelmointiin, vaikka opetuksessa käytettäisiin sellaista ohjelmointikieltä, joilla molemmat ovat mahdollisia (Sim ja Lau 2022). Samanlaisia tuloksia on havaittu myös muissa julkaisuissa (katso esim. Robins, J. Rountree ja N. Rountree 2003) ja täten ei ole yllättävää, että tekstipohjaisia ohjelmointikieliä käytettäessä Pythonia suositaan alkeis- ja peruskursseissa ja olio-ohjelmointiin nojaavaa Javaa puolestaan jatkokursseissa. Muita opetuksessa käytettäviä tekstipohjaisia ja olio-ohjelmointiin viittaavia ohjelmointikieliä ovat C, C++ ja C# (Ou et al. 2023). Myös Pythonilla voidaan opettaa olio-ohjelmointia (Saidova 2022), mutta muista kielistä poiketen Pythonin käyttö opetuksessa yksinään ei suoraan viittaa siihen, että opintojaksolla opetettaisiin olio-ohjelmointia.

Ohjelmoinnin opetukseen liittyy olennaisesti myös opetettavan sisällön ja ympäristön lisäksi opetuksessa käytetyt teknologiat. Ohjelmointia voidaan opettaa pe-

rinteisesti pelkästään tekstipohjaisilla ohjelmointikielillä ja käyttäen jotakin tekstiä käsittelevää koodieditoria, mutta yhtä hyvin voidaan ottaa mukaan fyysisiä elementtejä ja opettaa ohjelmoinnin taitoja robotiikan tai laajemmin elektroniikan kautta (Çınar ja Tüzün 2021; Fülöp et al. 2022). Robotiikka on esimerkki niin kutsutusta fyysisestä ohjelmoinnin opetuksesta, jolla on todettu olevan positiivinen vaikutus oppimistuloksiin. Toinen tällainen merkittävä ja tehokas opetusteknologia on visualisointi, josta mainio esimerkki ovat jo aiemmin mainitut palikkapohjaiset ohjelmointikielet. (Scherer, Siddiq ja Sánchez Viveros 2020.) Visualisointia voidaan myös käyttää tukemaan tekstipohjaisten ohjelmoinitkielten opetusta, kuten on tehty Kailan ja muiden tutkimuksessa (Kaila et al. 2018).

Vaikka ohjelmoinnille ei ole virallista globaalia didaktiikkaa, niin selvästi ympäri maailman on pyritty selventämään ohjelmoinnin opetuksen kenttää. Niinpä myös opetusmetodeista on kirjoitettu jonkin verran. Fuentes Martinez on tuoreessa väitöskirjassaan (Fuentes Martinez 2024) nostanut esille kolme erilaista metodia, joilla ohjelmointia yleisesti opetetaan tutkimuskirjallisuuden valossa. Metodeista ensimmäinen on demonstraatio, joka on varsin opettajakeskeinen opetusmetodi ja jossa ideana on, että oppilaat seuraavat opettajan demonstroimaa ohjelmakoodia ja oppivat tästä kokemuksesta. Yksi tapa tehokkaasti käyttää demonstraatioita on niin kutsuttu reaaliaikainen koodaus (eng. live coding), jossa opettaja kirjoittaa ohjelmakoodia oppilaiden edessä ja täten antaa realistisen kuvan ohjelmoinnista sekä oppilaille mahdollisuuden olla mukana koodausprosessissa. (Fuentes Martinez 2024, s. 22–23.) Reaaliaikaista ohjelmointia on myös suositeltu artikkelissa (Brown ja Wilson 2018). Toinen metodeista on vapaasti suomennettuna nikkarointi (eng. tinkering), jossa oppilaille annetaan ohjelmakoodia, jota heidän tulee tutkia ja muokata (Fuentes Martinez 2024, s. 23-24). Kyseessä on selkeästi demonstraatioita oppilaslähtöisempi opetusmetodi, sillä oppilailla on mahdollisuus kokeilla ja edetä melko vapaasti omaan suuntaan ja omaan tahtiin. Viimeinen nimetty metodi on ns. debuggaaminen, eli ohjelmakoodin virheiden paikannus ja korjaus. Samoin kuin nikkaroinnissa, opetuskontekstissa on tyypillistä, että oppilaille ainakin alkuvaiheessa annetaan näennäisesti valmista, mutta kuitenkin virheellistä ohjelmakoodia, mitä tutkia. Debuggaaminen on työelämässä arkipäiväistä, mutta samalla aloitteleville ohjelmoijille haastavaa. (Fuentes Martinez 2024, s. 26-28.)

2.2 Ohjelmoinnin opetus matematiikassa

Aivan kuten ohjelmoinnilta puuttuu yleisesti hyväksytty metodiikka, puuttuu myös yksi tietty oppiaine, jossa ohjelmointia opetetaan. Osittain tämä johtuu siitä, että ohjelmoinnille ja niille tieteenaloille, joille ohjelmointi on keskeistä, ei ole muodostu-

nut vakiintunutta termistöä. Toisin sanoen ei ole olemassa sellaista tieteenalaa eikä oppiainetta, jonka alle ohjelmointi ja sen opetus kuuluisivat. (Nojonen 2024, s. 4.) Tämä voidaan nähdä esimerkiksi eri maiden tavoissa sisällyttää ohjelmointi kansallisiin opetussuunnitelmiin 2010- ja 2020-luvuilla. Jako voidaan pääosin tehdä maihin, joissa ohjelmoinnin opetusta varten varattiin opetussuunnitelmassa paikka omalle, ohjelmointiin suoraan viittaavalle oppiaineelle, kuten tietotekniikalle, sekä maihin, joissa ohjelmoinnin opetus on integroitu täysin jo olemassa oleviin oppiaineisiin, kuten matematiikkaan. (Bocconi et al. 2022.)

Puuttuvan vakiintuneen termistön lisäksi yksi merkittävä syy oman oppiaineen tai integroidun lähestymistavan valinnassa on ohjelmoinnillisen ajattelun painoarvo. Tämä Jeanette Wingin (Wing 2006) popularisoima termi on saanut vuosien varrella lukuisia eri muotoja, mutta esimerkiksi Muhammadin ja muiden tuoreessa bibliometrisessä tutkimuksessa (Muhammad et al. 2024) ohjelmoinnillinen ajattelu on määritelty systemaattisena ja järjestelmällisenä tapana kehittää ratkaisuja ongelmiin. Lisäksi ohjelmoinnillisen ajattelun osa-alueet vaihtelevat hieman lähteiden välillä, mutta yleisesti ainakin abstraktio, yleistäminen, ongelman pilkkominen osiin, algoritmiikka tai algoritminen ajattelu sekä debugaaminen mielletään ohjelmoinnillisen ajattelun taitoina (Wing 2006; Balanskat ja Engelhardt 2015; Bocconi et al. 2022; Lv, Zhong ja Liu 2023; Muhammad et al. 2024). Ohjelmoinnillisen ajattelun taidot on määritelty tärkeiksi tulevaisuudessa menestymisen kannalta ja juuri tästä syystä näitä taitoja on pyritty sisällyttämään opetussuunnitelmiin (Balanskat ja Engelhardt 2015; Bocconi et al. 2022).

Ohjelmoinnillisen ajattelun taidot eivät välttämättä vaadi juuri tietokoneohjelmoinnin opetusta (Weintrop, Beheshti et al. 2016), vaan ohjelmoinnillista ajattelua voidaan pitää paljon laajempana taitona kuin pelkkää ohjelmointia — ohjelmointi vaatii ohjelmoinnillista ajattelua, mutta ohjelmoinnillinen ajattelu ei välttämät rajoitu pelkästään ohjelmointiin. Samalla, koska ohjelmoinnillinen ajattelu on paljon laajempi konsepti kuin ohjelmointi, on ohjelmoinnillisella ajattelulla merkittäviä yhteneväisyyksiä esimerkiksi yleisen ongelmanratkaisutaidon sekä matemaattisten taitojen kanssa (Barr ja Stephenson 2011, s. 117). Näin on siis perusteltu sitä, että välttämättä ei tarvita omaa oppiainetta, jossa ohjelmoinnillista ajattelua opetettaisiin. Samalla, koska matematiikka ja tietojenkäsittelytiede ovat muutenkin läheisiä tieteenaloja (katso esim. Gal-Ezer ja Harel 1998), tuntuu varsin loogiselta, että juuri matematiikkan opetukseen integroitaisiin ohjelmoinnillisen ajattelun opetus.

Miksi sitten ohjelmointia opetetaan matematiikan tunneilla, jos tavoitteena on ohjelmoinnillisen ajattelun kehittäminen ja tähän on olemassa myös muita keinoja? Tutkimuskirjallisuudesta ei löydy tähän yksiselitteistä vastausta, mutta osasyynä lie-

nee se, että työmarkkinoille on tarvittu tai nähty tarvittavan juuri ohjelmointia osaavia työntekijöitä (Balanskat ja Engelhardt 2015). Samalla tavoitteena on varmasti antaa realistisempi kuva matematiikasta, sillä ohjelmoinnin ja muiden tietoteknisten apuvälineiden käyttö työkaluina on alalla nykyisin arkipäivää (Weintrop, Beheshti et al. 2016, s. 128). Tieteellisestä näkökulmasta katsottuna on järkevää opettaa ohjelmoinnillista ajattelua varsinaisen ohjelmoinnin avulla ja Merino-Armeron, González-Caleron sekä Cózar-Gutiérrezin tekemän meta-analyysin mukaan ohjelmointi on tehokkain tapa oppia ohjelmoinnillista ajattelua (Merino-Armero, González-Calero ja Cózar-Gutiérrez 2022). Lisäksi Webb ja muut tietojenkäsittelyn ja ohjelmoinnillisen ajattelun opetuksen merkitystä käsittelevässä artikkelissaan kirjoittavat, että näiden aiheiden alkeiskurssi ilman ohjelmointia olisi käsittämätön idea (Webb et al. 2017, s. 457). Täten, vaikka opetussuunnitelmassa päätavoitteena olisi opettaa ohjelmoinnillista ajattelua matematiikassa, tarkoittaa tämä yleensä samalla sitä, että myös ohjelmointia opetetaan matematiikan oppiaineessa.

Vaikka ohjelmointi pohjautuu matematiikkaan ja molemmissa käytetään samoja, tai ainakin hyvin samankaltaisia, ajattelun taitoja, ohjelmoinnin opetuksen sisällyttäminen matematiikkaan ei välttämättä takaa matemaattisten taitojen kehittymistä. Esimerkiksi Kauffmannin ja Stensethin artikkelissa (Kaufmann ja Stenseth 2021) todetaan, että ohjelmoinnin sisällytys matematiikan opetukseen ei välttämättä suoraan vahvista oppilaiden ongelmanratkaisutaitoja. Samassa artikkelissa todetaan myös, että jotta ohjelmoinnista olisi hyötyä matematiikan opetuksessa, tulisi opettajalla olla vahvat ohjelmointitaidot. Toisin sanoen, jos halutaan ohjelmoinnin avulla vahvistaa ohjelmoinnillista ajattelua sekä matemaattisia taitoja, ei pelkän ohjelmoinnin lisääminen matematiikan opetukseen riitä, vaan ohjelmoinnin opetuksen tulee olla tarkoituksenmukaista ja nojata vahvaan opettajan osaamiseen.

Siihen, miten ohjelmoinnin opetus integroidaan matematiikkaan, on olemassa erilaisia lähestymistapoja. Tämän tutkielman kannalta erittäin mielenkiintoisia ovat Fuentes Martinezin väitöskirjassaan (Fuentes Martinez 2024) kuvailemat opettajien taktiikat. Tämä johtuu pääosin siitä, että Fuentes Martinezin väitöskirjan tutkimus on tehty Ruotsissa, missä tilanne ohjelmoinnin opetuksen suhteen on hyvin samankaltainen kuin Suomessa. Väitöskirjassa on esitetty kaksi eri taktiikkaa, joilla opettajat ovat pyrkineet sisällyttämään opetussuunnitelmareformin takia ohjelmointia matematiikkaan. Nämä kaksi taktiikkaa ovat duaaliopetus (eng. dual teaching) sekä siroteltu ohjelmointi (eng. interspersed programming).

Duaaliopetuksessa opettajat erottelevat ohjelmoinnin ja matematiikan omiksi kokonaisuuksiksi, joista matematiikalla on selvästi suurempi painoarvo. Tässä opetustaktiikassa ohjelmointi nähdään opetettavana asiana, joka on osittain irrotettu

matematiikasta ja seuraa omaa suunnitelmaansa. (Fuentes Martinez 2024, s. 101-106.) Esimerkiksi jos ensin matematiikan kokonaisuudessa on opiskeltu kolme oppituntia suorakulmaisten kolmioiden geometriaa, niin neljännellä oppitunnilla saatetaan vaihtaa ohjelmointiin, opiskella muuttujia sekä funktioita ja ehkä kirjoittaa Pythagoraan lausetta hyödyntävä ohjelma. Näin tultaisiin kuitenkin tekemään vain silloin, jos matematiikassa on edetty tarpeeksi pitkälle ja opettajan suunnitelmasta löytyy niin sanotusti vapaata aikaa. Toisin sanoen duaaliopetuksessa matematiikka ja ohjelmointi on erotettu toisistaan siten, että pääpaino on aina matematiikalla ja ohjelmointia voidaan opettaa tämän lisäksi, kunhan se ei tule matematiikan tielle.

Sirotellussa ohjelmoinnissa puolestaan ohjelmointi nähdään matematiikan työkaluna, jota voidaan satunnaisesti hyödyntää oppitunneilla, kun opettaja näkee sen tukevan matematiikkaa. Ero duaaliopetukseen on selvä, sillä sirotellussa ohjelmoinnissa ohjelmointia ei nähdä omana opetettavana kokonaisuutena, vaan pelkästään matematiikkaa tukevana apuvälineenä. Tällöin ohjelmointi on myös pitkälti vapaaehtoista, eli kun ohjelmointia käytetään jonkin matemaattisen ongelman ratkaisussa, niin yhtä hyvin oppilas voi hyödyntää jotakin muuta matemaattista keinoa ja olla ohjelmoimatta. Lisäksi sirotellussa ohjelmoinnissa ohjelmointi seuraa tarkalleen matematiikan kokonaisuuden suunnitelmaa, eli mikäli matemaattisessa kokonaisuudessa ei ole opettajan mielestä järkevää käyttää ohjelmointia, niin hyvin todennäköisesti ohjelmointia ei tässä tilanteessa käytetä laisinkaan. (Fuentes Martinez 2024, s. 106-111.)

Näyttää siis vahvasti siltä, että tällä hetkellä ohjelmoinnin opetuksen integrointi matematiikan opetukseen tuottaa haasteita. Fuentes Martinez mainitsee väitöskirjassaan (Fuentes Martinez 2024) yhdeksi merkittäväksi syyksi sen, että lukioissa on rajattu määrä oppitunteja käsitellä matematiikan osa-alueita ja kun opettajat ajattelevat oppilaidensa parasta, on silloin tärkeämpää keskittyä kokeissa kysyttäviin teemoihin kuin ohjelmointiin. Samalla suuri vaikutus on sillä, että Ruotsissa ohjelmoinnille tai tietotekniikalle ei ole olemassa ylioppilastutkintoon johtavaa koetta, eli ohjelmointi ei ole valmistumisen kannalta oleellista. Samanlainen havainto on tehty myös Suomessa ja Karvin vuoden 2022 raportissa selkeä syy ohjelmoinnin vähäiselle suosiolle lukiotasolla oli nimenomaan ylioppilaskokeen puuttuminen (Nousiainen ja Kivistö 2022).

Matematiikan ja ohjelmoinnin opetuksen yhteensovittamiselle on myös olemassa muitakin kuin koulutuspoliittisia haasteita. Esimerkiksi Alegren ja muiden artikkelissa (Alegre et al. 2020) todetaan, että matematiikan ja ohjelmoinnin yhteensovittaminen on haastavaa ja yksi suuri syy tähän on ohjelmointikielten vaatima syntaksi. Syntaksin vaikeus ja epäselvyys eivät ole ominainen ongelma ohjelmoinnin

integroidulle opetukselle, vaan yleinen haaste ohjelmoinnin opetuksessa (Bau et al. 2017). Kuitenkin yhdistettynä rajallisen oppituntimäärän ja sen faktan kanssa, että matematiikalla on usein paljon suurempi painoarvo, ei ole yllättävää, että opettajat tinkisivät ohjelmoinnin opetuksesta ainakin tekstipohjaisilla ohjelmointikielillä. Yen ja muiden kirjallisuuskatsauksessa onkin mainittu, että ohjelmoinnin opetusta voidaan matematiikan opetuksessa toteuttaa jopa taulukkolaskentasovellusten, kuten Excelin, kanssa, mitä ei tietenkään voida pitää ohjelmointina vaan enintään ohjelmoinnillista ajattelua tukevana työtapana (Ye et al. 2023, s. 11). Ongelmat syntaksin kanssa eivät kuitenkaan ole ylitsepääsemättömiä ja esimerkiksi jos opettaja kokee ohjelmoinnin syntaksin hankaloittavan ohjelmoinnin opetusta, niin palikkaohjelmointi on varteenotettava vaihtoehto. Tai kuten Alegre ja muut artikkelissaan (Alegre et al. 2020) ehdottavat, opetuksessa voidaan myös käyttää jotakin tekstipohjaista ohjelmointikieltä, jonka syntaksi on joko yleisesti kevyempää tai lähempänä matematiikassa käytettäviä merkintöjä. Joka tapauksessa, kun tarkastellaan ohjelmoinnin opetusta integroituna muihin oppiaineisiin ja erityisesti matematiikkaan, on tärkeää pysyä kriittisenä ja pyrkiä tulkitsemaan milloin kyseessä on todella ohjelmoinnin opetusta ja milloin ohjelmointia opetetaan omana, tärkeäksi koettuna, kokonaisuutenaan eikä ainoastaan työkaluna, kuten viivainta tai harppia.

2.3 Ohjelmoinnin opetus Suomessa

Koska tämän tutkielman keskiössä on Suomen lukioissa järjestettävä opintojakso, on mielekästä tarkastella laajempaa suomalaista kontekstia ja sitä, miten ohjelmointia opetetaan lukioissa sekä sitä edeltävässä koulutuksessa. Suomi, kuten monet muut Euroopan maat, sisällytti ohjelmointia kansallisiin opetussuunnitelmiinsa ensimmäistä kertaa 2010-luvulla, kun vuoden 2014 peruskoulun opetussuunnitelman perusteisiin (POPS) kirjattiin mainintoja ohjelmoinnista (Opetushallitus 2014). Tässä tällä hetkellä tuoreimmassa POPSissa on ohjelmointi integroitu täysin muihin oppiaineisiin ja vielä tarkemmin matematiikkaan sekä käsitöihin. Ohjelmointi on osana laaja-alaisen osaamisen tavoitteita kaikilla perusopetuksen käsittävillä vuosiluokilla ja mainitaan esimerkiksi vuosiluokkien 7–9 tieto- ja viestintäteknologian osaamisen tavoitteessa (L5) seuraavasti: ”Ohjelmointia harjoitellaan osana eri oppiaineiden opintoja.” Käsityön oppiaineessa ohjelmointi on läsnä 7–9 vuosiluokan keskeisissä sisältöalueissa siten, että osana opintoja tulisi tutustua sulatettuihin järjestelmiin ja tätä kautta saadaan tarttumapinta ohjelmointiin (Opetushallitus 2014, s. 431).

Kuitenkin valtaosa ohjelmoinnin opetuksesta on integroitu matematiikan oppiaineeseen ja kaikkien vuosiluokkien matematiikan tavoitteista löytyy ohjelmointia jonkin verran. Vuosiluokilla 1—2 tutustutaan ohjelmoinnin alkeisiin ohjelmoimatta ja

vuosiluokilla 3–6 laaditaan ohjelmia graafisissa ohjelmointiympäristöissä, eli todennäköisesti palikkaohjelmointia hyödyntämällä. Vuosiluokilla 7–9 ohjelmointi mainitaan osana algoritmista ajattelua ja opetellaan ehtolauseiden käyttöä, ”hyviä ohjelmointikäytäntöjä” sekä yleisesti kirjoitetaan yksinkertaisia ohjelmia ja käytetään jo olemassa olevia ohjelmia matemaattisten ongelmien ratkaisemiseen. POPS2014 ei määritä, millä ohjelmointikielillä ohjelmointia opetetaan tai mitkä ovat käytännön oppimistavoitteet ehtolauseiden lisäksi. Opettajien työtä tukevasta materiaalista, kuten Järvenpään oppikirjasta (Järvenpää 2019), voidaan kuitenkin tulkita, että todennäköisesti 7–9 luokkalaiset käyttävät ohjelmoinnissa Pythonia ja että ohjelmointia opetetaan joitakin tunteja lukuvuodessa. Yhden oppikirjan perusteella jää kuitenkin epäselväksi, kuinka paljon ohjelmointia todellisuudessa opetetaan, mutta opetussuunnitelmatekstin perusteella voidaan sentään todeta, että ohjelmoinnin opetusta on olemassa ja ohjelmoinnin tulisi olla jollain tasolla tuttua oppilaille, kun he siirtyvät lukioon.

Lukioihin ohjelmointi saapui vasta vuoden 2019 lukion opetussuunnitelman perusteissa (LOPS). Ennen tätä tällä hetkellä tuoreinta LOPSia ohjelmointia ei ollut mainittu laisinkaan eikä lukiotasolle ollut valtakunnallisesti lainkaan ohjeistusta ohjelmoinnin opetukseen. LOPS2019 muutti tilannetta hieman, kun pitkän matematiikan moduuliin MAA11, jossa käsitellään algoritmeja ja lukuteoriaa, sisällytettiin ohjelmoinnin opetusta. Opetussuunnitelmassa oleva kuvaus on kuitenkin varsin väljä. Ohjelmointi mainitaan kahdesti MAA11 kuvaavassa tekstissä siten, että moduulin tavoitteissa oppilaan tulisi oppia ”toteuttamaan yksinkertaisia algoritmeja ohjelmoimalla” ja moduulin keskeisissä sisällöissä mainitaan: ”yksinkertaisten algoritmien, lajittelualgoritmien tai yhtälön numeeriseen ratkaisuun liittyvän algoritmin ohjelmointi.” (Opetushallitus 2019, s. 228-229.) Toisin sanoen ohjelmointi on selvästi läsnä opetussuunnitelmassa, mutta ei tarjoa minkäänlaista osviittaa siitä, mitä tämä ohjelmointi käytännössä tarkoittaa.

Jos tutkitaan ohjelmoinnin opetusta puhtaasti opetussuunnitelmatasolla, niin on perusteltua tehdä tulkinta, että MAA11 ei sisällä näiden kahden maininnan pohjalta merkittävissä määrin ohjelmoinnin opetusta (Nojonen 2024, s. 19). Tämä ei kuitenkaan tarkoita, että ohjelmoinnin opetusta ei olisi lainkaan. Esimerkiksi kun LOPS2019 julkaistiin, tarjosi Matemaattisten Aineiden Opettajien Liitto (MAOL) opettajille tukimateriaalia, jonka avulla opetussuunnitelman tulkitseminen helpottui. Tässä oheismateriaalissa mainitaan MAA11 moduulin kohdalla: ”Ohjelmointi toteutetaan jollakin ohjelmointikielellä, esimerkiksi Pythonilla. Opetussuunnitelman tavoitteita ei voi saavuttaa pelkällä taulukkolaskennalla” (Matemaattisten Aineiden Opettajien Liitto 2019, s. 33). Toisin sanoen vaikka itse opetussuunnitelmateks-

ti jättää pimentoon, miten ohjelmoinnin opetus toteutetaan, niin oheismateriaalin mukaan tarkoituksena olisi varsinaisen ohjelmoinnin opetus jollakin tekstipohjaisella ohjelmointikielellä.

Myös kaikkein yleisimmin käytetyt oppikirjat viittaavat siihen, että MAA11 todella sisältää ohjelmoinnin opetusta. Sekä Otavan Juuri 11 (Hähköniemi et al. 2022) että Sanoma Pron Moodi 11 (Harsunkorpi et al. 2023) sisältävät kokonaisen kappaleen, joka on omistettu ohjelmoinnille. Molemmissa kirjoissa käytetään LOPS2019 tukimateriaalissa mainittua Pythonia ohjelmointikielenä ja molemmissa käydään läpi ohjelmoinnin perusteita, kuten ehto- ja toistolauseita. Ei kuitenkaan voida suoraan olettaa, että ohjelmoinnin opetus vastaisi näissä oppikirjoissa esitettyä tasoa. Ylioppilastutkinnossa ei ole erikseen ohjelmoinnin osuutta (Ylioppilastutkintolautakunta 2024) ja täten saattaa hyvinkin olla, että opettaja tekee oppimateriaalien ja opetussuunnitelman perusteella sellaiset tulkinnat, jotka johtavat joko duaalio-petuksen tai sirotellun ohjelmoinnin kaltaisiin taktiikoihin. Tällöin oppimateriaali kyllä viittaa siihen, että kaikille MAA11 suorittaneille oppilaille on opetettu ohjelmoinnin perusteet, mutta realiteetti saattaa olla toisenlainen. Täten, kun halutaan realistinen kuva ohjelmoinnin opetuksesta integroituna MAA11 moduuliin, ei voida tehdä tulkintoja pelkästään virallisten dokumenttien, kuten LOPS2019 tai sen tukimateriaalin, tai vastaavasti yleisimpien oppimateriaalien perusteella.

3 Tutkimuskysymykset, tutkimuksen tarkoitus ja tutkimusmenetelmät

Tämän tutkimuksen päätavoitteena on selvittää ohjelmoinnin opetuksen tila valtakunnallisessa MAA11 moduulissa, sillä se on tällä hetkellä ainoa lukiotason opintojakso, jossa ohjelmointi mainitaan opetussuunnitelman perusteissa. Tässä tutkielmassa pyritään tarkentamaan, mitä MAA11 moduulissa mainittu ohjelmoinnin opetus käytännössä tarkoittaa, sillä pelkän opetussuunnitelmatekstin perusteella jää hyvin epäselväksi, kuinka suuri painoarvo ohjelmoinnilla on ja mitä opetuksen tulisi pitää sisällään. Täten mielenkiintona eivät ole ainoastaan opetuksessa käytetyt teknologiat tai mitä ohjelmointikieliä opetuksessa käytetään, vaan myös miten opettajat suhtautuvat ohjelmoinnin opetukseen. On siis tärkeä selvittää, käytetäänkö ohjelmoinnin opetuksessa esimerkiksi sirotellun ohjelmoinnin tai duaaliopetuksen kaltaisia taktiikoita, tuottavatko oppilaat opintojaksolla itsenäisesti ohjelmakoodia vai seuraavatko he ainoastaan opettajan näyttämiä esimerkkejä.

Aineistonkeruu on toteutettu sähköisen kyselylomakkeen avulla, eli tutkielma on toteutettu kyselytutkimuksena. Tutkielmassa hyödynnetään määrällisiä metodeja, sillä tarkoituksena on tuottaa tilastollista tietoa esimerkiksi käytetyistä opetusteknologioista ja siitä, miten opettajat suhtautuvat ohjelmointiin matematiikan oppiaineessa ja onko tällä vaikutusta siihen, miten ohjelmointia opetetaan.

Tutkimuskysymykset ovat:

1. Miten ohjelmointia opetetaan MAA11 opintojaksolla?
2. Mitä teknologioita ohjelmoinnin opetuksessa käytetään MAA11 opintojaksolla?

3.1 Tutkimusaineisto

Tutkimusaineisto on kerätty ja käsitelty Turun yliopiston ylläpitämällä REDCap-ohjelmistolla (Harris, Taylor, Thielke et al. 2009; Harris, Taylor, Minor et al. 2019). REDCap (Research Electronic Data Capture) on turvallinen, selainpohjainen ohjelmistoalusta, joka on suunniteltu tukemaan tutkimusdatan keräämistä sähköisesti. Tähän tutkielmaan REDCap valikoitui erityisesti sen vuoksi, että REDCapin avulla on suoraviivaista luoda sähköinen kyselylomake ja välittää se eteenpäin vastaajille, sekä siksi, että REDCap mahdollistaa kerätyn tiedon saumattoman viennin erilaisiin ulkopuolisiin ohjelmiin, joilla datan varsinainen analyysi toteutetaan.

Kyselylomakkeessa A on yhteensä yhdeksän kohtaa, joista kaksi ensimmäistä (A1-2) ovat avokysymyksiä ja toimivat yksittäisen lomakkeen avainarvoina. Kysy-

mys A6 on yhden vaihtoehdon valinta ja loput kuusi (A3-5 ja A7-9) ovat monivalintoja, joilla kartoitetaan ohjelmoinnin opetusta MAA11 opintojaksolla. Lomakkeessa ei kerätä vastaajien henkilötietoja ja vain hyvin harvoissa tapauksissa voitaisiin yksittäinen vastaus yhdistää tiettyyn opettajaan ja tällöinkin ainoastaan todennäköisesti. Esimerkiksi jos vastaus on tullut lukiosta, jossa toimii ainoastaan yksi pitkän matematiikan opettaja, ei tätä tietoa ole välttämättä julkisesti saatavilla ainakaan ottamatta erikseen yhteyttä oppilaitokseen. Mahdollisuus yksittäisen henkilön tunnistamiseen on kuitenkin olemassa, ja tämän pienen todennäköisyyden vuoksi tässä tutkielmassa ei tulla esittelemään vastauksia koulukohtaisesti tai mainitsemaan nimeltä yksittäisiä lukioita, vaikka ne poikkeavaisitkin merkittävästi muista lukioista.

Kyselylomakkeen kysymykset A3, A6 ja A7 pyrkivät selkeyttämään ja mittaamaan ohjelmoinnin merkitystä opintojaksolla. Tarkemmin A3 pyrkii varmistamaan, että opintojaksolla ylipäätään opetetaan ohjelmointia, A6 kartoittaa, suhtautuuko vastaaja ohjelmointiin opintojaksolla enemmän duaaliopetuksen vai sirotellun ohjelmoinnin kaltaisella taktiikalla ja A7 tarkentaa, ohjelmoivatko oppilaat itse opintojakson aikana ja yhdessä kysymyksen A3 kanssa antavat viitteitä siitä, kuinka paljon oppilaat ohjelmoivat opintojakson aikana. Kysymykset A4, A5 ja A8 käsittelevät ohjelmoinnin opetuksen toteutusta ja sisältöä opintojaksolla siten, että A4-5 keskittyvät siihen, mitä ohjelmoinnista opetetaan opintojaksolla ja puolestaan A8 miten ohjelmointia opetetaan. Lopuksi kysymys A9 antaa tietoa opintojaksolla käytetyistä oppimateriaaleista ja mahdollisuuden vertailla esimerkiksi opetettuja ohjelmistotekniikoita siihen, mitä useimmin käytetyissä oppimateriaaleissa on kirjattuna.

Tutkimuslomake esitestattiin kahden pitkän matematiikan opettajan kanssa, minkä jälkeen se lähetettiin sähköpostitse yhteensä 298 lukioon joko rehtorille, apulaisrehtorille tai jollekin matematiikan opettajalle riippuen siitä, mitä yhteystietoja yksittäisen lukion verkkosivuilla oli saatavilla. Sähköpostissa pyydettiin vastaanottajaa jakamaan kysely kyseisen lukion pitkän matematiikan opettajille. Lista lukiosta saatiin opintopolku.fi sivulta hakemalla kaikki ne lukiot, jotka tarjoavat ylioppilastutkintoon johtavaa koulutusta maksuttomana päiväopetuksena. Tämän jälkeen etsittiin jokaisen lukion verkkosivut tai vastaavasti kuntien verkkosivut, joissa ylläpidetään lukion henkilökunnan yhteystietoja. Kysely lähetettiin alun perin lukioille toukokuussa 2024 ja vastauksia kerättiin koko toukokuun ajan. Kyselyyn saatiin 74 vastausta yhteensä 61 eri lukiosta, eli noin 20% kaikista kyselyn saaneista lukioista vastasi kyselyyn.

3.2 Analyysi

Kyselyn vastausten analyysi on toteutettu siten, että RedCapin tuottama tietue on käsitelty taulukkolaskentasovelluksella sellaiseen muotoon, josta on ollut mahdollista lukea jokaisen palautetun vastauslomakkeen tiedot numeerisesti. Koska vastauslomakkeet ovat tunnistettavissa niille annetun korttinumeron perustella, viitataan yksittäisiin lomakkeisiin ja sen vastauksiin korttinumeron perusteella, jotta vastauksen antaja pysyy anonyyminä.

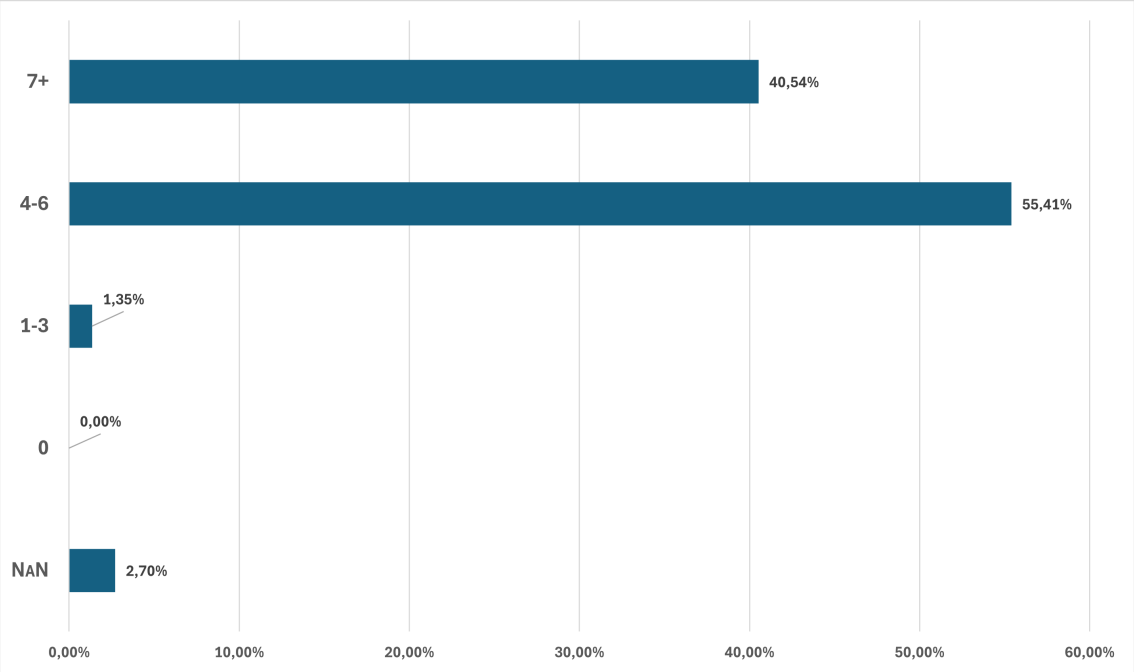
Tietueen muokkauksen jälkeen on laskettu annettujen vastausten suhteelliset osuudet kunkin kysymyksen kohdalla sekä otettu talteen sellaiset korttinumerot, jotka poikkeavat jollakin tavalla muusta aineistosta. Esimerkiksi jos yli 95 prosenttia korteista on vastannut kahden vaihtoehdon välillä, on nämä loput viisi prosenttia kirjattu ylös, jotta voidaan pitää kirjaa siitä, jos samat kortit poikkeavat myös muiden vastausten kohdalla. Tämän yleisen analyysin jälkeen kaikki kysymykset on käyty uudelleen läpi lomakkeen kysymyksen A3 vastauksen perusteella ja näin tarkasteltu oppituntimäärän vaikutusta muihin muuttujiin.

Tämän jälkeen analyysia on jatkettu tarkastelemalla muiden kuin oppituntimäärien vaikutusta toisten kysymysten vastauksiin niissä tilanteissa, joissa aiemman kahden kierroksen aikana on havaittu jotakin mielenkiintoista, mistä on haluttu saada lisätietoa. Mikäli muuttujien välinen suhde on näyttänyt jollakin tapaa poikkeavalta tai mielenkiintoiselta, on niiden välillä laskettu Pearsonin korrelaatiokerroin r, joka on vaikuttavuudeltaan nimetty joko pieneksi $(0,1 < |r| < 0,3)$, keskisuureksi $(0,3 < |r| < 0,5)$ tai suureksi $(0,5 < |r|)$ Cohenin määritelmien mukaan (Cohen 1988, s. 75–83). Yleisesti ottaen korrelaatiokertoimet ovat kuitenkin joko mitättömiä tai hyvin pieniä ja täten ne on mainittu ainoastaan silloin, jos korrelaatio muuttujien välillä on vähintään keskisuuri tai jos suurista poikkeavaisuuksista huolimatta korrelaatiokerroin ei vastaa havaintoa. Kaikkien vastausten ja niiden välisten suhteiden analysoinnin jälkeen on vielä tarkasteltu jokaisessa vaiheessa ylösotettuja poikkeavia korttinumeroita keskenään ja katsottu, ovatko samat kortit olleet poikkeavia useassa vaiheessa vai ainoastaan tietyissä kohdissa.

4 Vastausten analyysi

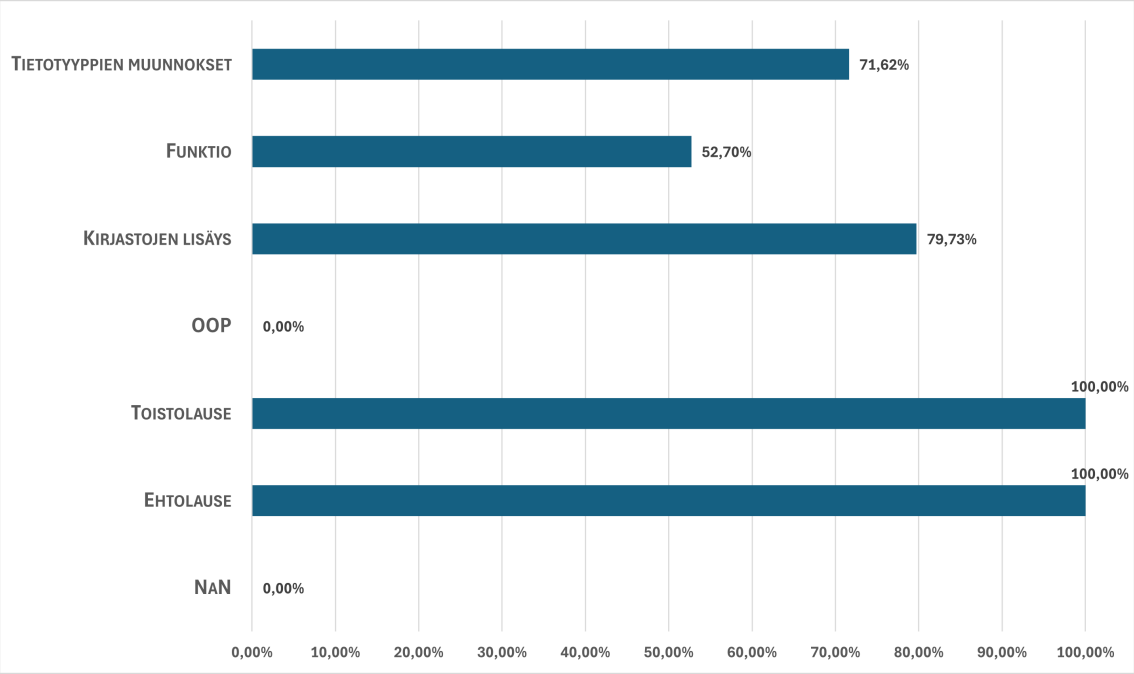
Lähes kaikissa kyselyyn vastanneissa lukioissa MAA11 opintojaksolla opetetaan ohjelmointia vähintään yhdellä oppitunnilla. Ainoastaan kahdessa lukiossa, korttinumerot 53 ja 65, jätettiin lomakkeen kysymykseen A3 vastaamatta, mutta molemmissa tapauksissa kaikkiin myöhempiin kohtiin oli vastattu huolellisesti. Todennäköisesti vastausta ei ole annettu joko epähuomiossa tai kyseisissä lukioissa ei ole järjestetty MAA11 opintojaksoa hiljattain ja täten opettajat ovat vastanneet omien suunnitelmiensa mukaisesti. Lisäksi ainoastaan yhdessä lukiossa, korttinumero 59, on vastaukseksi annettu 1-3 oppituntia, eli miltei kaikissa vastauksen antaneissa lukioissa ohjelmointia opetetaan vähintään neljällä 75 minuutin oppitunnilla tai vastaavasti kuudella 45 minuutin oppitunnilla.

Vastaukset vaikuttavat loogisilta, sillä esimerkiksi Juuri 11 -oppikirjassa on ohjelmoinnin kokonaisuudelle suositeltu kuusi 75 minuutin oppituntia tai kymmenen 45 minuutin oppituntia. Vastaukset jakaantuivat siten, että noin 55 prosentissa kaikista lukioista ohjelmointia käsitellään neljästä kuuteen oppitunnilla ja noin 41 prosentissa yli seitsemällä. Koska nämä kaksi kategoriaa ovat selvästi suurimmat, on mielekästä tutkia pääasiassa niiden välisiä eroja, kun halutaan selvittää oppituntimäärän vaikutusta muiden kysymysten vastauksiin.



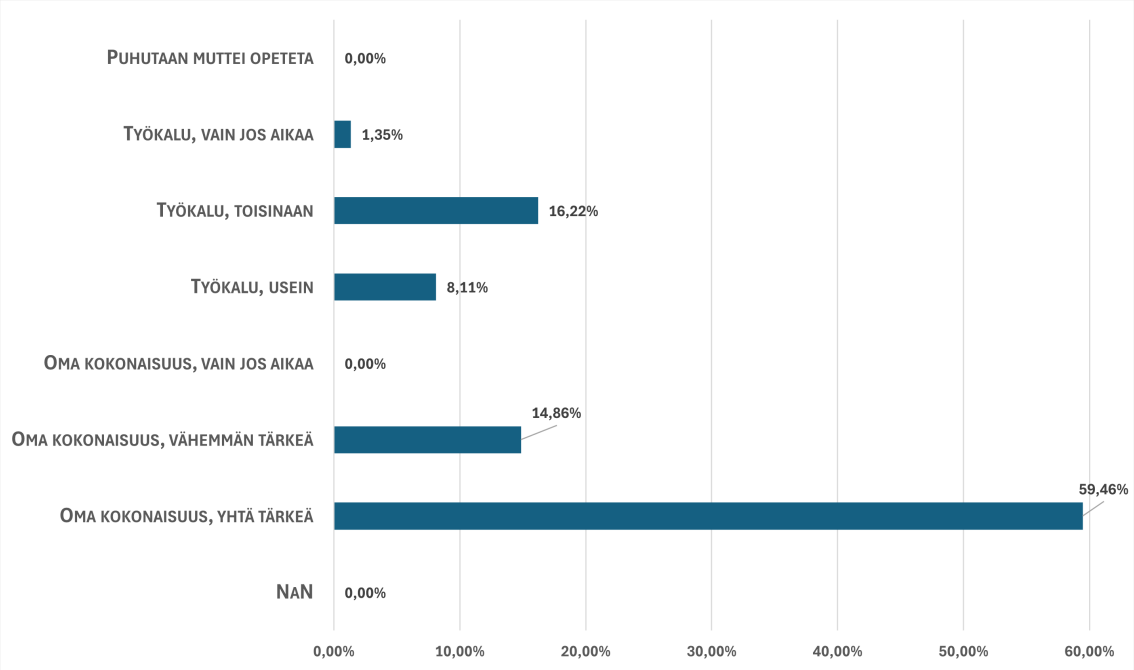
Kuva 1: 75 minuutin oppituntien lukumäärien suhteelliset osuudet kaikissa vastauksissa kysymykseen A3. NaN tarkoittaa, ettei kyseiseen kysymykseen ole vastattu laisinkaan.

Opetuksessa käytetyt teknologiat heijastavat opetussuunnitelman tavoitteita. Kaikissa 74 vastauksessa on mainittu, että kurssin aikana opeteltaisiin ehtolauseiden ja toistolauseiden käyttöä. Nämä kaksi ovat oleellisia LOPS2019 asettamissa tavoitteissa, sillä MAA11 kohdalla keskeisissä sisällöissä mainitaan ”algoritmisen ajattelun peruskäsitteet: peräkkäisyys, valinta ja toisto.” Eli koska näitä teknologioita käsitellään opintojaksolla yleisesti matematiikan kokonaisuudessa, niin on luonnollista, että ne ovat keskiössä ohjelmoinnin kokonaisuudessa. Näiden lisäksi kirjastojen lisäys ja tietotyyppien muunnokset on mainittu yli 70 prosentissa vastauksista ja funktio yli puolessa. Ohjelmoinnin osuudessa ei siis näytetä keskittyvän pelkästään matematiikassa esillä olleisiin teknologioihin, vaan opetukseen on otettu mukaan myös sellaisia aiheita, jotka ovat oleellisia nimenomaan ohjelmoinnissa. Olio-ohjelmointia ei ole mainittu yhdessäkään vastauksessa, mikä viittaa siihen, että MAA11 opintojaksolla opetetaan ainoastaan ohjelmoinnin perusteita. Tämä ei ole yllättävää, sillä kaikissa vastauksissa on ohjelmointikieleksi mainittu Python, joka ei ole tunnettu olio-ohjelmoinnin periaatteita painottavana kielenä. Pythonin lisäksi ainoat muut vastauksissa annetut ohjelmointikielet ovat Excel, GeoGebra sekä LibreCalc, joista Excel ja LibreCalc ovat puhtaasti taulukkolaskentasovelluksia ja GeoGebra matematiikassa käytetty graafinen piirtotyökalu ja Computer Algebra System (CAS). Myöskään näissä sovelluksissa ei yhdessäkään hyödynnetä olio-ohjelmointia.



Kuva 2: Erilaisten opetettavien teknologioiden suhteelliset osuudet kaikissa vastauksissa kysymykseen A5. NaN tarkoittaa, ettei kyseiseen kysymykseen ole vastattu laisinkaan. OOP on olio-ohjelmoinnista yleisesti käytetty lyhenne.

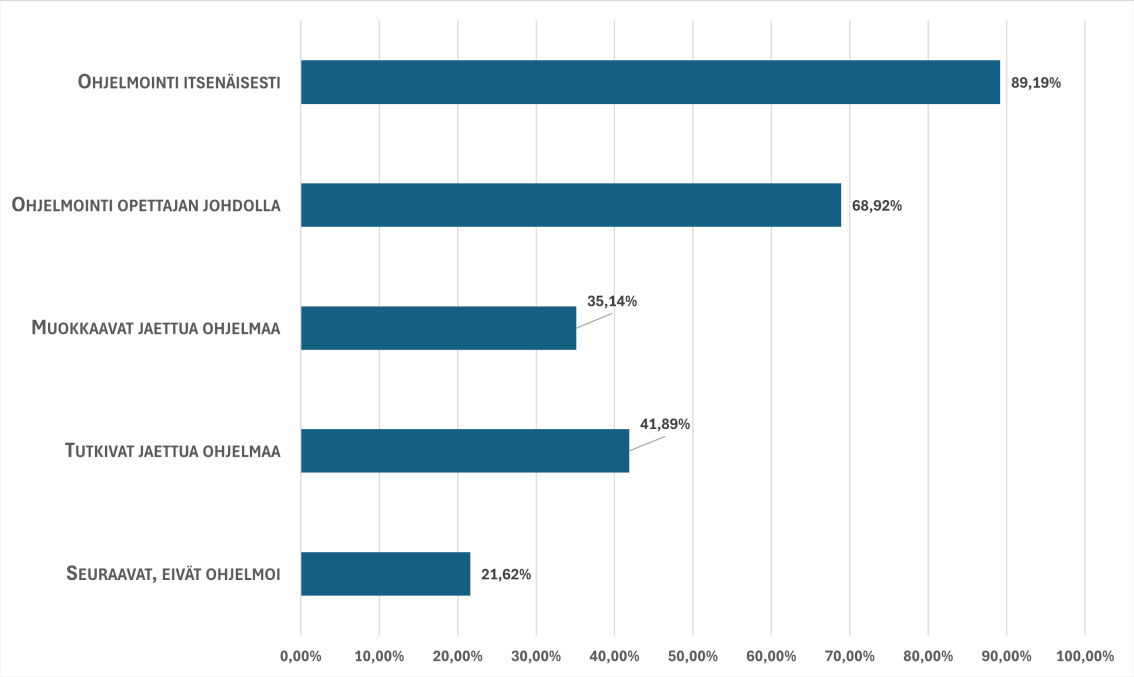
Opettajien suhtautuminen ohjelmointiin MAA11 kurssilla on erittäin mielenkiintoisesti jakautunut. Selvästi yli puolessa vastauksista, eli noin 75 prosentissa, ohjelmointia pidetään omana opetettavana kokonaisuutena. Täten ainoastaan neljänneksessä vastauksista ohjelmointi nähdään pikemminkin matematiikassa käytettävänä työkaluna. Yhteensä noin 60 prosentissa vastauksista ohjelmoinnin kokonaisuus nähdään yhtä tärkeänä kuin matematiikan kokonaisuus ja ainoastaan 15 prosentissa vähemmän tärkeänä. Eli mikäli ohjelmointia opetetaan omana kokonaisuutena, näyttäisi sen arvostus olevan melko suurta matematiikkaan verrattuna. Puolestaan jos ohjelmointi on mielletty työkaluksi, niin ohjelmoinnin merkitys näyttää olevan vähäisempi. Ainoastaan kahdeksassa prosentissa vastauksista ohjelmointi on usein käytettävä työkalu ja noin kahdeksassatoista prosentissa ohjelmointia käytetään joko toisinaan tai vain, jos siihen jää aikaa. Yhdessäkään koulussa ei kuitenkaan vastausten perusteella sivuuteta ohjelmoinnin opetusta ainoastaan puhumalla aiheesta, vaan oppilaat todella pääsevät kirjoittamaan tai ainakin näkemään ohjelmakoodia.



Kuva 3: Ohjelmoinnin merkitys opintojaksolla suhteessa matematiikkaan kaikissa vastauksissa kysymykseen A6. NaN tarkoittaa, ettei kyseiseen kysymykseen ole vastattu laisinkaan.

Vastauksissa mainituissa työtavoissa löytyy jonkin verran vaihtelua. Melkein kaikissa, eli noin 90 prosentissa vastauksista, on mainittu, että oppilaat ohjelmoivat itsenäisesti opintojakson aikana. Niissä noin kymmenessä prosentissa vastauksista, joissa oppilaat eivät ohjelmoi itsenäisesti, pääsevät oppilaat kuitenkin ohjelmoimaan opettajan johdolla. Täten kaikkien vastausten mukaan ei yhdessäkään koulussa jäte-

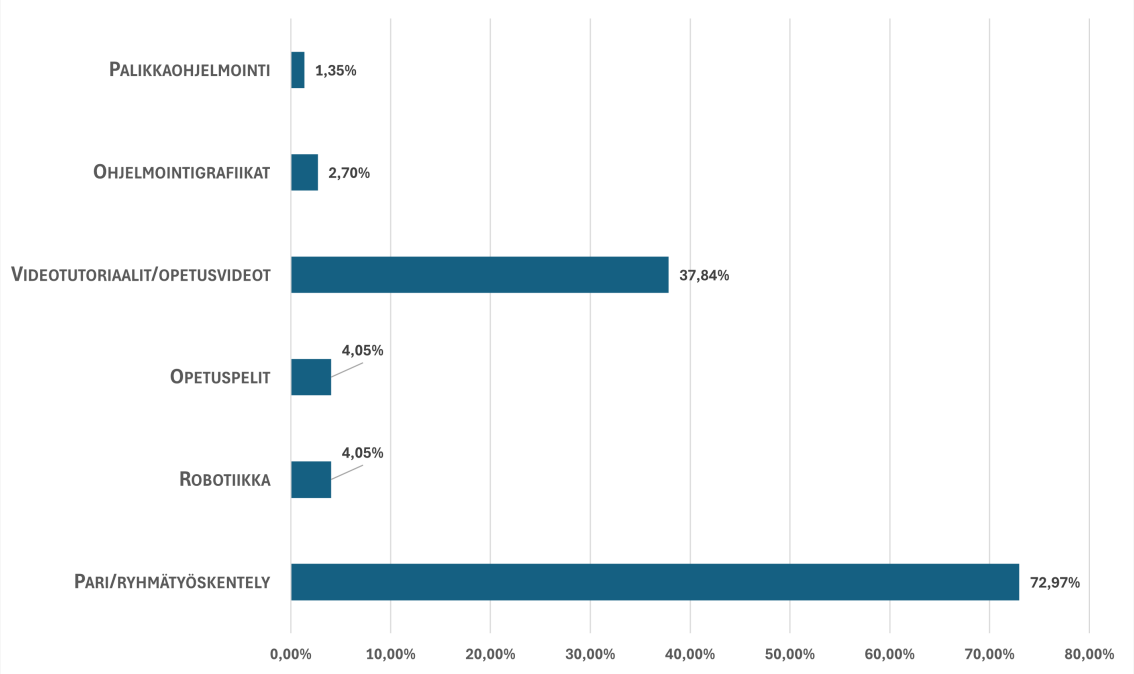
tä ohjelmointia pelkästään opettajan näyttämien esimerkkien varaan, vaan oppilaat pääsevät ainakin jossain kohtaa opintojaksoa myös kirjoittamaan ohjelmakoodia. Mielenkiintoista on myös, että hieman alle puolessa vastauksista, eli noin 47 prosentissa, oppilaille jaetaan opintojakson aikana valmiiksi kirjoitettua ohjelmakoodia, jota oppilaat sitten tutkivat tai muokkaavat. Sinänsä ei ole yllättävää, että ohjelmakoodin jakaminen oppilaille on melko suosittu opettamistapa – onhan nikkarointi yleisesti tunnistettu opetusmetodi (Fuentes Martinez 2024, s. 23-24). Kuitenkin se, että näin moni matematiikan opettaja on kokenut hyödylliseksi luoda itse ohjelmakoodia opintojaksoa varten, vaikka ohjelmointia ei vaadita matematiikan opettajan tutkintoon tai pätevyyteen, on positiivinen yllätys.



Kuva 4: Miten ohjelmoinnin opetus toteutettu opintojaksolla kaikissa vastauksissa kysymykseen A7.

Vastausten perusteella MAA11 kurssilla käytetään verrattain vähän ohjelmoinnin opetuksen tutkimuskirjallisuudesta löytyneitä erilaisia opetusmetodeja. Suuressa osassa ohjelmoinnin opetuksessa hyödynnetään pari- tai ryhmätyöskentelyä, mutta muita metodeja mainitaan selvästi alle puolessa vastauksista. Videotutoriaalit ja opetusvideot ovat myös melko suosittuja ja ne on mainittu yli kolmanneksessa vastauksista, mutta muista metodeista löytyy vain yksittäisiä esiintymiä. Tämä ei ole yllättävä tulos, sillä esimerkiksi MAA11 opintojaksolle tarkoitetuissa oppikirjoissa, kuten Juuri 11 tai Moodi 11, ei ole hyödynnetty varsinaisesti yhtään kysymyksen A8 vastausvaihtoehdoista. Samalla kaikissa vastauksen antaneissa kouluissa käytetään opetuskielenä Pythonia, mikä ei suoranaisesti liity esimerkiksi palikkaohjelmointiin.

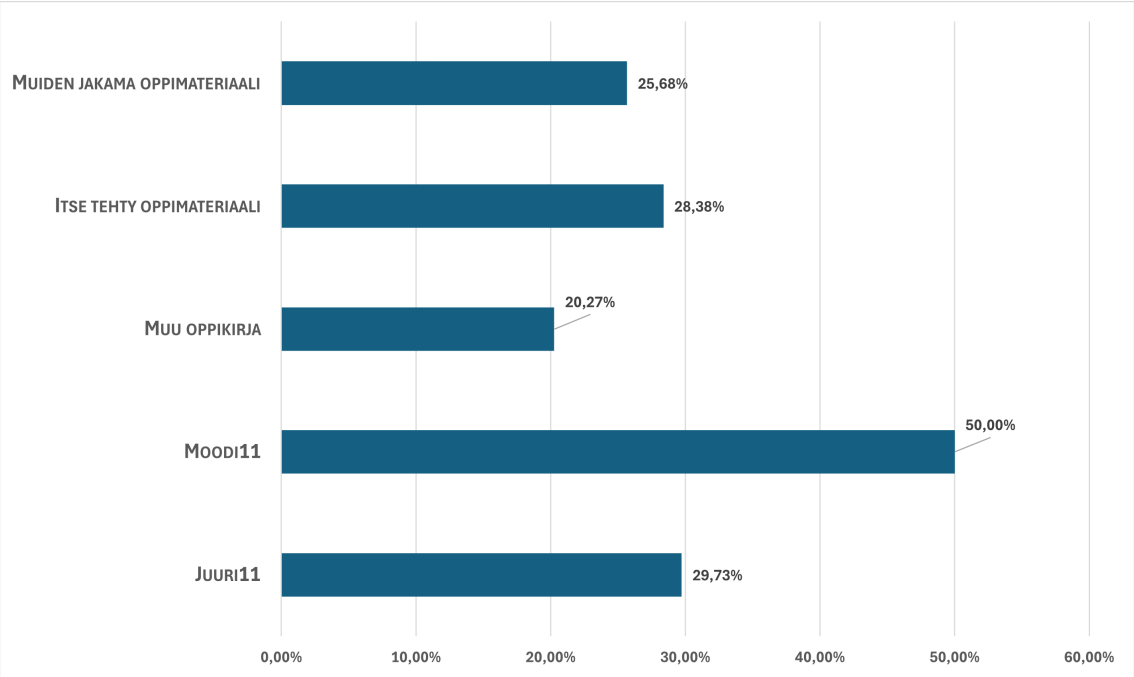
Mielenkiintoista onkin, että löytyy ylipäätään sellaisia kouluja, joissa matematiikan kurssilla opetetaan ohjelmointia esimerkiksi robotiikan avulla, sillä tämä eroaa merkittävästi perinteisestä matematiikan opetuksesta. Yllättävin tulos opetusmetodeihin liittyen on, että ohjelmointigrafiikoita hyödynnetään hyvin vähän. Ne on todettu yhdeksi parhaiksi tavoiksi opettaa ohjelmointia aloittelijoille, mutta vastausten perusteella opetusmetodia hyödynnetään vain kahdessa lukiossa. Vastausten pientä määrää voi selittää esimerkiksi huono kysymyksenasettelu eivätkä vastaajat ole tarkalleen tienneet, mitä ohjelmointigrafiikoilla on tarkoitettu. Vastaavasti voi myös olla, että opetus pyritään mahdollisesti pitämään mahdollisimman lähellä perinteistä matematiikan opetusta, eikä täten ohjelmoinnin osuuteen ole erikseen luotu graafisia elementtejä.



Kuva 5: Ohjelmoinnin opetuksessa käytettyjen opetusmetodien suhteelliset osuudet kaikissa vastauksissa kysymykseen A8.

Oppimateriaaleja käytetään melko vaihtelevasti. Noin puolessa vastauksista mainitaan SanomaPron kustantama Moodi 11, mikä on selvästi suosituin yksittäinen oppikirja tämän kyselyn vastauksissa. Puolestaan Otavan kustantama Juuri 11 on mainittu noin 30 prosentissa vastauksista. Yhdessäkään vastauksessa ei ole mainittu, että koulussa käytettäisiin sekä Moodi 11:tä ja Juuri 11:tä samanaikaisesti, eli noin 80 prosentissa kaikista vastauksista on käytössä jompikumpi näistä suosituista oppikirjoista. Ainoastaan yhdessä lukiossa, korttinumero 49, ei ole mainittu lainkaan oppikirjaa oppimateriaalina. Toisin sanoen tässä lukiossa opintojaksoa opetetaan pelkästään itse tehtyjen tai muiden jakamien oppimateriaalien perusteella.

Miltei kaikissa lukiossa on siis käytössä jokin oppikirja. Opintojaksoa ei kuitenkaan joissain lukiossa opeteta pelkästään yksittäisen oppikirjan sisällön perusteella, vaan jopa noin 43 prosentissa vastauksista on lisäksi mainittu, että opetuksessa käytetään joko itse tehtyä tai muiden jakamaa oppimateriaalia.



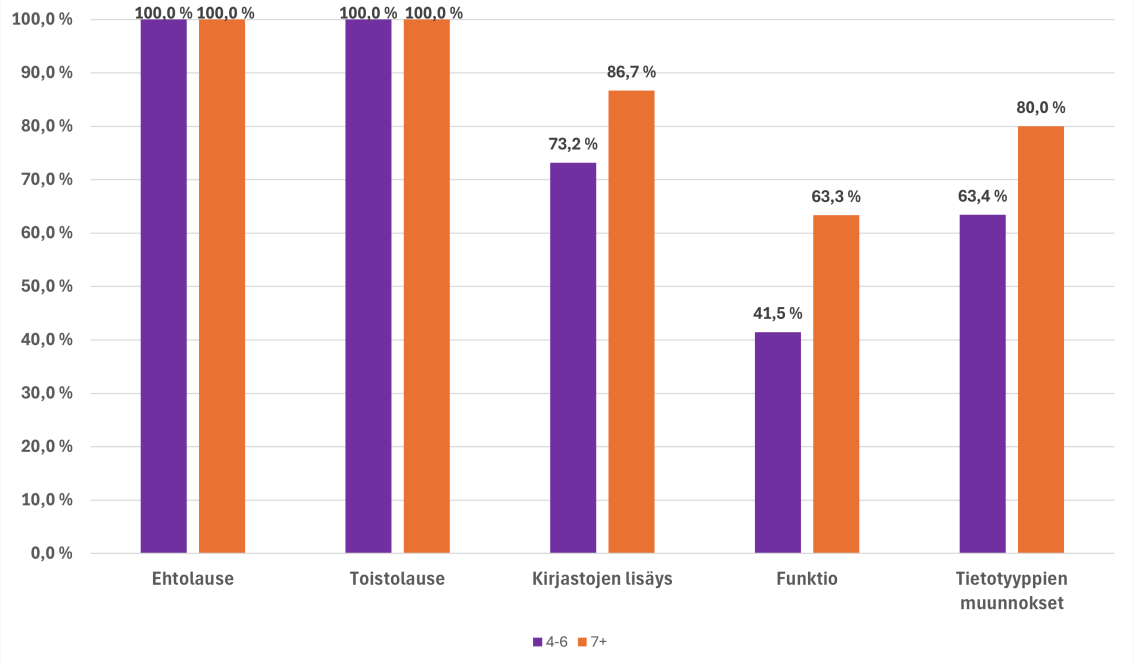
Kuva 6: Erilaisten oppimateriaalien suhteelliset osuudet kaikissa vastauksissa kysymykseen A9.

4.1 Oppituntimäärän vaikutus muihin muuttujiin

Koska kaikissa vastauksen antaneissa lukiossa käytetään Pythonia ohjelmointikielenä, jää oppituntimäärän merkitys muihin lomakkeen kysymysten vastauksiin mielenkiintoisimmaksi tarkemman analyysin kohteeksi. Oletettavasti mitä enemmän oppitunteja opintojaksolla käytetään, sitä enemmän opetusmetodeja, opetettavia teknologioita ja erilaisia opetustoteutuksia vastauksissa esiintyy. Samalla oletettavasti ohjelmoinnin merkitys koetaan useimmin vähintään yhtä tärkeäksi matematiikan kokonaisuuden kanssa. Koska oppituntimäärät ovat vastauksissa jakaantuneet siten, että yli 95 prosenttia korteista ovat vastanneet joko 4–6 tai 7+ oppituntia, ovat näiden kahden ryhmän väliset erot kaikkein mielenkiintoisimmat. Tästä syystä myös suurin osa kaavioista tulee käsittelemään ainoastaan näitä kahta ryhmää ja havainnot kolmesta poikkeustapauksesta, kortit 53, 59 ja 65, esitellään suurelta osin vain tekstiosuuksissa, jos ne ovat jollain tavalla mielenkiintoisia ja oleellisia.

Oletusten mukaisesti oppituntien määrällä näyttäisi olevan vaikutus opintojak-

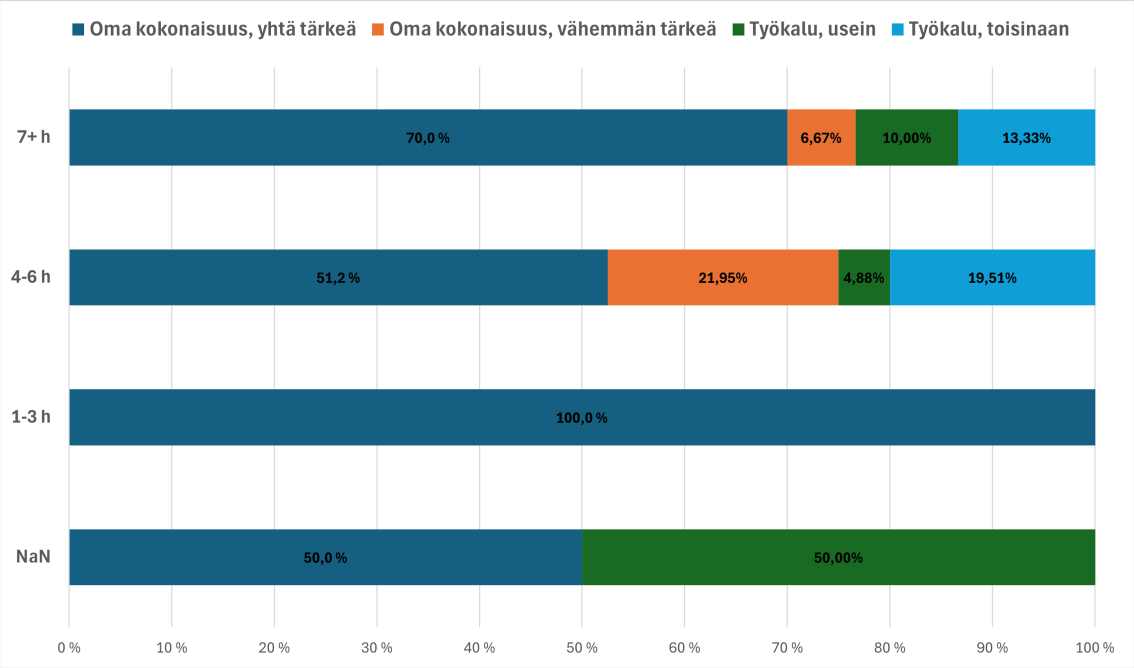
son aikana opetettaviin teknologioihin. Kaikissa vastauksissa on mainittu ehto- ja toistolauseet, mutta muut kolme vastauksissa esiintynyttä teknologiaa ovat jonkin verran yleisempiä niissä korteissa, joissa kysymykseen A3 on vastattu 7+, kuin niissä, joissa vastaus on 4–6. Tämä on loogista, sillä mitä enemmän oppitunteja opintojaksolla käytetään ohjelmointiin, sitä enemmän jää aikaa käydä läpi erilaisia ohjelmointiin liittyviä teknologioita ja tekniikoita. Tämä ei kuitenkaan tarkoita, että mahdollisesti pienemmillä määrillä oppitunteja ei voisi käydä läpi yhtä paljon teknologioita ja tekniikoita. Kaikissa poikkeustapauksissa on jokaisessa mainittu kaikki mahdolliset teknologiat pois lukien olio-ohjelmointi.



Kuva 7: Erilaisten opetettavien teknologioiden suhteelliset osuudet niissä korteissa, joissa vastaus kysymykseen A3 on ollut joko 4–6 tai 7+

Niissä kouluissa, joissa vastausten mukaan opetetaan ohjelmointia enemmän, myös arvostetaan ohjelmointia hieman enemmän. Vastauksissa on enemmän sellaisia kouluja, joissa MAA11 sisältää 4–6 oppituntia ohjelmointia, kuin niitä, joissa oppitunteja on yli seitsemän. Kuitenkin molemmissa ryhmissä oli absoluuttisesti yhtä paljon sellaisia vastauksia, joissa ohjelmointia pidetään omana ja matematiikan kanssa yhtä tärkeänä kokonaisuutena. Samoin niiden korttien määrä, joissa ohjelmointia pidettiin usein käytettävänä työkaluna, oli suurin piirtein yhtä paljon. Tämä näkyy suhteellisissa osuuksissa siten, että yli seitsemän oppitunnin kategoriassa ohjelmoinnin merkitys on paljon suurempi, kuin 4–6 kategoriassa. Tuloksissa tulee kuitenkin huomioida, että 4–6 oppitunnin kategoria ei välttämättä ole homogeeninen ja saattaa olla, että esimerkiksi neljällä ja kuudella oppitunnilla on huomattava

ero siihen, kuinka tärkeänä ohjelmointia pidetään. Tästä riippumatta näyttää kuitenkin selvältä, että mitä enemmän oppitunteja MAA11 opintojaksolla käytetään ohjelmointiin, sitä tärkeämpänä ohjelmointi nähdään suhteessa matematiikkaan.

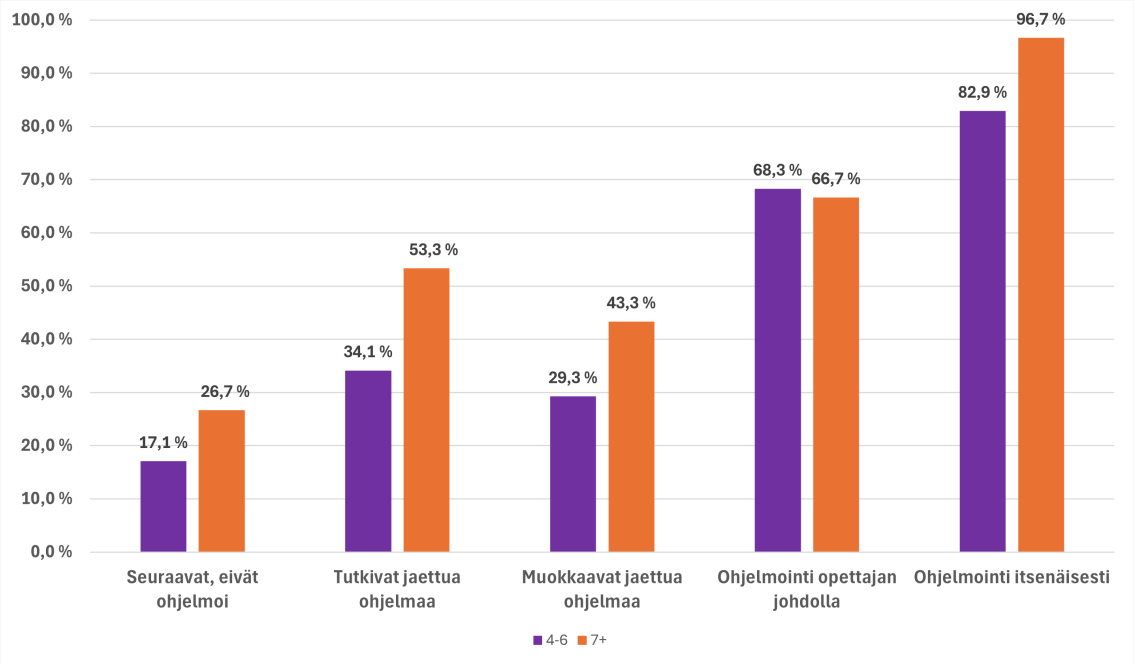


Kuva 8: Ohjelmoinnin merkityksen suhteelliset osuudet oppituntien määrän mukaan. NaN tarkoittaa, että oppituntimäärää ei ole annettu.

Myös opetustavoista voidaan nähdä, että mitä enemmän ohjelmointia sisällytetään MAA11 opintojaksolle, sitä enemmän oppilaat myös ohjelmoivat. Lähes kaikissa niissä lukioissa, joissa ohjelmointiin käytetään vähintään seitsemän oppituntia, ohjelmoivat oppilaat itsenäisesti ainakin jonkin verran. Puolestaan jos oppitunteja on neljästä kuuteen, on itsenäinen ohjelmointi silti yleinen opetusmetodi, mutta ei läheskään niin käytetty, kuin niissä korteissa, joissa ohjelmointia sisältäviä oppitunteja on vähintään seitsemän. Samoin on vastausten perusteella paljon todennäköisempää, että opettaja jakaa oppilaille ohjelmakoodia, jota tutkitaan tai muokataan, jos oppitunteja on vähintään seitsemän. Ainoa toteutustapa, joka on suositumpi niiden lukioiden keskuudessa, joissa oppitunteja on enintään kuusi, on ohjelmointi opettajan johdolla.

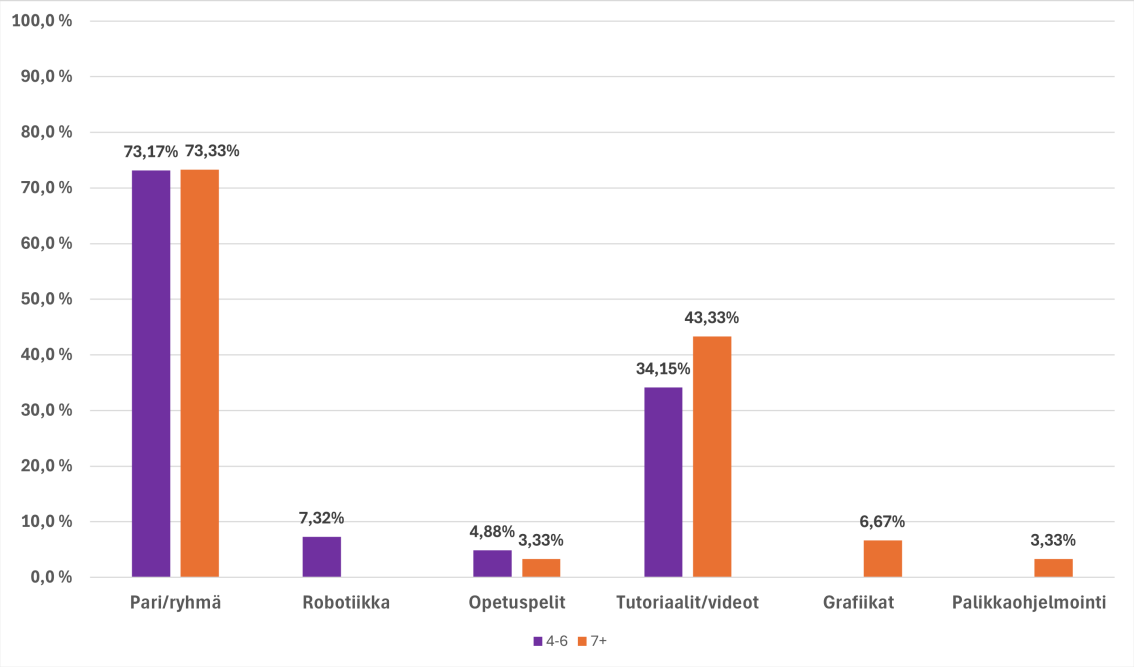
Yleisesti vastauksista saatiin, että kaikissa lukioissa oppilaat joko ohjelmoivat itsenäisesti tai opettajan johdolla, mutta eivät ainoastaan seuraa opetusta tai nikkaroi koodin kanssa. Saattaa siis olla, että mikäli opettaja ei ole syystä tai toisesta halunnut hyödyntää itsenäistä ohjelmointia, on kuitenkin nähty tärkeäksi, että oppilaat pääsevät kirjoittamaan ohjelmakoodia. Kuvan 9 kaavion perusteella voidaan olettaa, että oppilaat eivät pääse ohjelmoimaan itsenäisesti opintojaksolla, ellei tälle ole

tarpeeksi aikaa. Itsenäinen ohjelmointi on kuitenkin verrattain yleistä, vaikka ohjelmoinnin oppitunteja olisi alle seitsemän. Esimerkiksi kortissa 59, jossa oppitunteja vain yhdestä kolmeen, on myös mainittu opetustapana itsenäinen ohjelmointi. Täten oppituntien määrä ei missään nimessä ole ainoa tekijä, joka vaikuttaa opetuksen toteutukseen, mutta sillä saattaa kuitenkin olla jonkinlainen vaikutus.



Kuva 9: Erilaisten opetustoteutusten suhteelliset osuudet niissä korteissa, joissa vastaus kysymykseen A3 on ollut joko 4–6 tai 7+

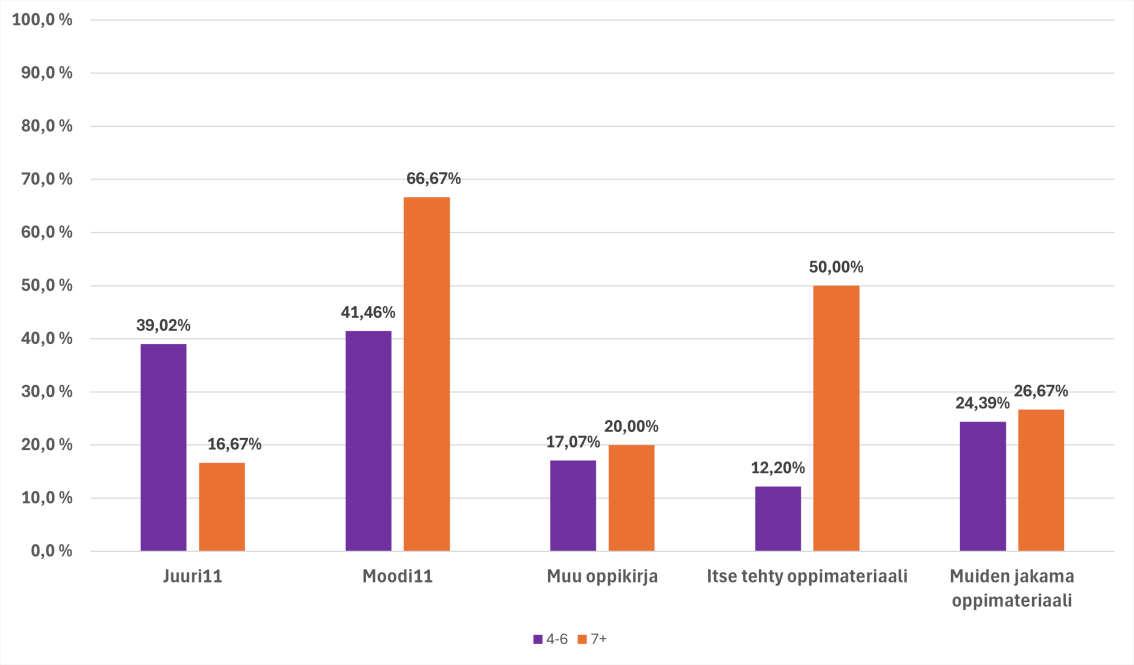
Oppituntien määrällä ei näyttäisi olevan juuri lainkaan vaikututusta opintojaksolla käytettäviin opetusmetodeihin. Ainoastaan opetustutoriaalien ja -videoiden kohdalla on havaittavissa, että suurempi oppituntimäärä saattaa lisätä niiden käyttöä. On kuitenkin mielenkiintoista, että robotiikkaa on käytetty ainoastaan niissä lukioissa, joissa ohjelmoinnin oppitunteja on 4–6 ja puolestaan ohjelmointigrafiikoita ja palikkaohjelmointia niissä, joissa oppitunteja on vähintään seitsemän.



Kuva 10: Erilaisten opetusmetodien suhteelliset osuudet niissä korteissa, joissa vastaus kysymykseen A3 on ollut joko 4–6 tai 7+

Oppimateriaaleissa näkyy mielenkiintoinen ero sen perusteella, onko kysymykseen A3 vastattu 4–6 oppituntia vai 7+ oppituntia. Juuri 11 näyttäisi olevan hyvin vähän käytössä, jos oppitunteja on vähintään seitsemän ja puolestaan Moodi 11 huomattavasti enemmän. Tämä voi olla selitettävissä sillä, että Moodi 11 sisältää hieman enemmän ohjelmointiin keskittyviä kappaleita kuin Juuri 11. Juuri 11:ssä ohjelmointia käsitteleviä kappaleita on yhteensä viisi, joista viimeinen on tarkoitettu oppilaiden tekemälle harjoitustyölle. Oppikirjassa näille viidelle kappaleelle on annettu suosituksekseksi kuusi 75 minuutin oppituntia. Moodi 11:ssä puolestaan ohjelmointia käsitteleviä kappaleita on kuusi ja kaikki on tarkoitettu ohjelmoinnin opetukseen. Moodi 11 ei anna oppituntimäärälle suosituksia, mutta oletettavasti koska sisältöä on hieman enemmän, olisi suosituksena myös hieman enemmän oppitunteja, kuin Juuri 11:ssä. Korrelaatiot oppituntimäärien ja oppikirjojen välillä ovat kuitenkin pieniä eikä tämän tutkielman havaintojen perusteella voida varmasti sanoa, että oppikirjan valinta automaattisesti vaikuttaisi oppituntien määrään.

Toinen merkittävä ero on, että itse tehty oppimateriaali on paljon yleisempää, jos oppitunteja on yli seitsemän. Tämä tarkoittaa mahdollisesti sitä, että oppikirjan materiaali saadaan valmiiksi noin kuuden oppitunnin kuluessa, mutta jos opettaja haluaa syventyä oppilaiden kanssa ohjelmointiin vielä enemmän, tarvitaan tätä varten opettajan itse luomaa materiaalia. Oppituntimäärän ja itse tehdyn oppimateriaalin välillä on myös keskisuuri korrelaatio, sillä korrelaatiokerroin 4–6 oppitunnin kohdalla on $r \approx -0,4$ ja 7+ oppitunnin kohdalla $r \approx 0,4$.



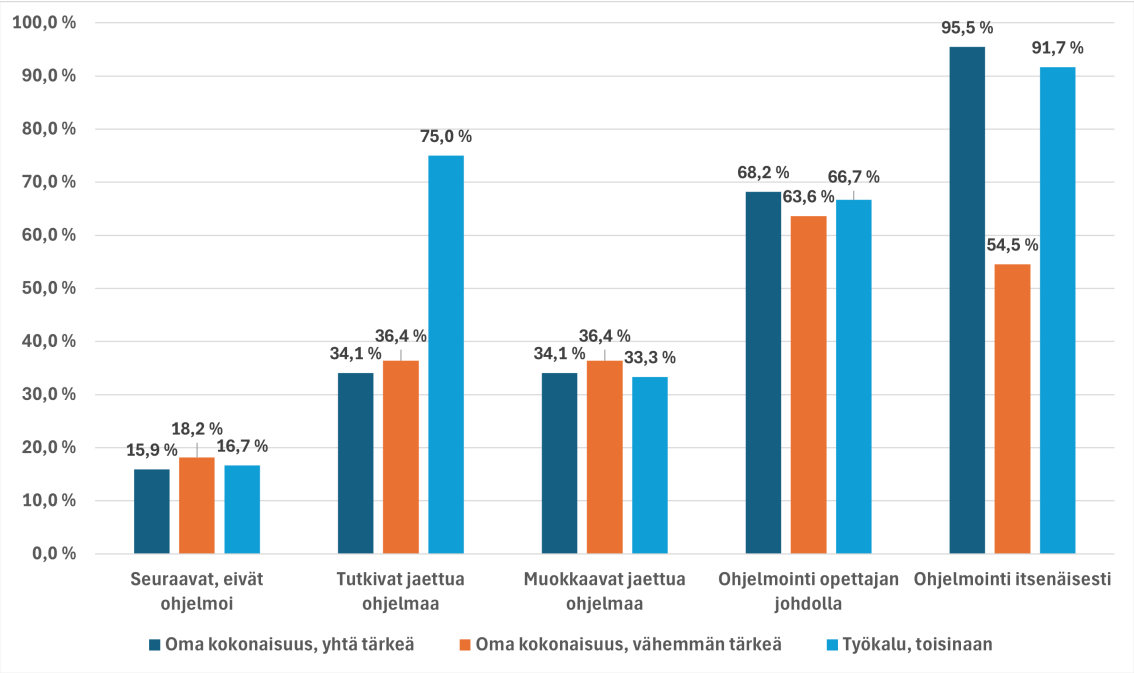
Kuva 11: Eri oppimateriaalien suhteelliset osuudet niissä korteissa, joissa vastaus kysymykseen A3 on ollut joko 4–6 tai 7+

4.2 Muita muuttujien välisiä havaintoja

Myös muiden vastausten välillä on havaittavissa mielenkiintoisia yhteyksiä. Kuvassa 12 nähdään ohjelmoinnin merkityksen ja opintojaksolla käytettyjen opetustoteutusten välinen suhde. Voidaan heti huomata, että suurelta osin sillä, miten ohjelmointi nähdään opintojaksolla, ei ole vaikutusta siihen, miten opetusta toteutetaan. Löytyy kuitenkin kaksi poikkeusta, jotka ovat yllättäviä. Ensinnäkin niissä korteissa, joissa ohjelmointi nähdään toisinaan käytettävänä työkaluna, on paljon yleisempää, että opintojakson aikana oppilaat tutkivat opettajan heille jakamaa ohjelmaa. Sama tulos on havaittavissa myös niissä korteissa, joissa ohjelmointi nähdään usein käytettävänä työkaluna. Tämä saattaa viitata siihen, että ohjelmointia käytetään opintojaksolla siten, että oppilaat tutkivat jotakin valmista ohjelmakoodia, jota käytetään

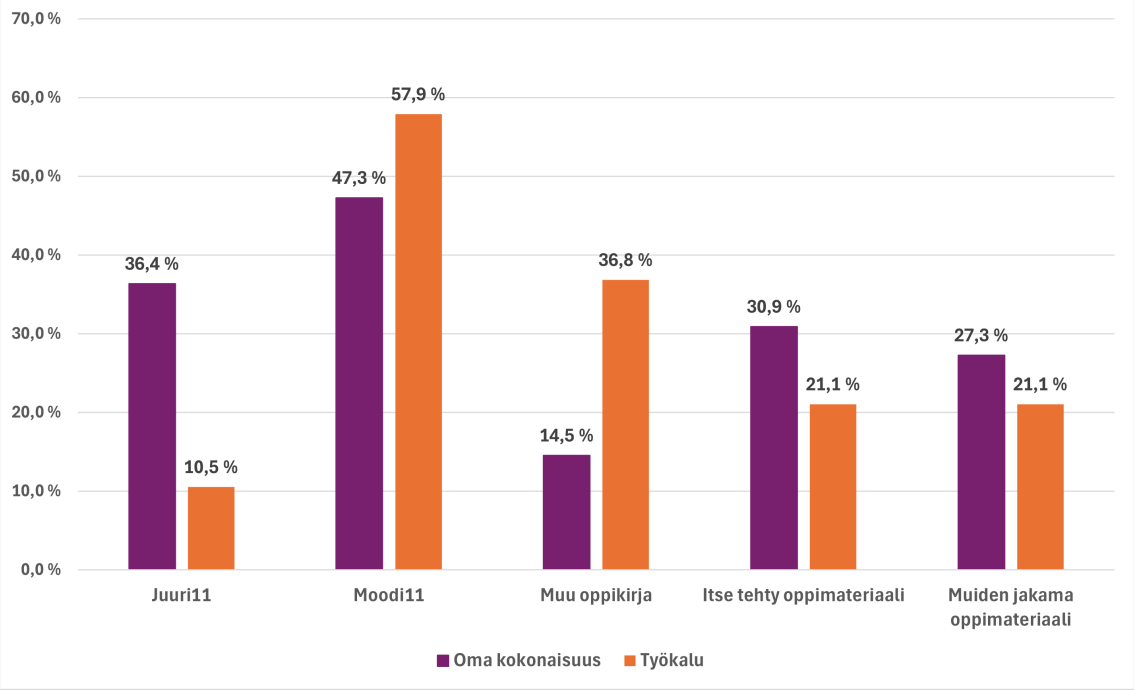
apuvälineenä joidenkin laskujen laskemisessa. Tämä selittäisi sen, että vaikka oppilaat tutkivat huomattavasti enemmän heille jaettua ohjelmaa, he eivät kuitenkaan muokkaa ohjelmaa yhtään sen enempää, kuin jos ohjelmointi on nähty omana kokonaisuutena. Korrelaatiokertoimien tarkastelu paljastaa, että jaetun ohjelman tutkimisella ja sillä, että ohjelma nähdään toisinaan käytettävänä työkaluna, on olemassa pieni korrelaatio ($r \approx 0,295$), mikä ei täysin vastaa Kuvassa 12 havaittavaa merkittävää eroa. Täytyy kuitenkin pitää mielessä, että vastausten lukumäärä on pieni eikä voida olla varmoja siitä, yleistyisikö tulos koko Suomen tasolle.

Toinen merkittävä poikkeus on, että itsenäinen ohjelmointi on paljon harvinaisempaa niissä korteissa, joissa ohjelmointi nähdään omana kokonaisuutena, joka on vähemmän tärkeä kuin matematiikka. Tämä on yllättävä tulos, sillä jos katsotaan kuvaa 9, voidaan havaita, että oppituntien määrällä ei ole suurta vaikutusta siihen, hyödynnetäänkö opintojaksolla itsenäistä ohjelmointia. Tämän lisäksi analyysi on tehty kaikkien muiden muuttujien suhteen, mutta ainoa itsenäiseen ohjelmointiin merkittävästi vaikuttava tekijä on, jos ohjelmointi nähdään omana, vähemmän tärkeänä kokonaisuutena. Myös korrelaatiokertoimet paljastavat samat havainnot ja ainoa merkittävä korrelaatio on nimenomaisesti itsenäisellä ohjelmoinilla ja jos ohjelmointi nähdään vähemmän tärkeänä, mutta omana kokonaisuutena ($r \approx -0,47$). Korrelaatiokerroin on myös yksi kaikkein suurimmista minkään kahden muuttujan välillä. Miksi näin on, ei ole selkeää. Kyseessä saattaa olla vain melko pienen aineistokoon aiheuttama sattuma, mutta yhtä hyvin voi olla, että jos opettaja käyttää duaaliopetuksen kaltaista strategiaa, ei hän tällöin koe, että itsenäinen ohjelmointi edesauttaisi opintojakson tavoitteiden saavuttamista.



Kuva 12: Erilaisten toteutustapojen suhteelliset osuudet ohjelmoinnin merkityksen mukaan. Kuvaan on selkeyden vuoksi otettu mukaan ainoastaan ne vaihtoehdot, joissa kysymykseen A6 on saatu vähintään kymmenen vastausta.

Tarkastelun arvoista on myös ohjelmoinnin merkityksen sekä käytettyjen oppimateriaalien välinen suhde. Vastausten perusteella Moodi 11 on yleisin oppikirja ja se on suosittu riippumatta siitä, miten ohjelmoinnin merkitys nähdään suhteessa matematiikkaan. Jos ohjelmointi nähdään omana kokonaisuutena, yhtä tai vähemmän tärkeänä kuin matematiikka, on tällöin paljon todennäköisempää, että oppikirjana on Juuri 11 kuin jos ohjelmointi nähdään työkaluna. Puolestaan, jos ohjelmointi nähdään työkaluna, on tällöin paljon todennäköisempää, että käytössä on jokin muu oppikirja, kuin jos ohjelmointi nähdään omana kokonaisuutena. Korrelaatiokertoimista voidaan tehdä samansuuntaisia tulkintoja. Jos ohjelmointi nähdään omana kokonaisuutena, on tällä pieni positiivinen korrelaatio Juuri 11 kanssa ja puolestaan pieni negatiivinen korrelaatio muun oppikirjan kanssa. Päinvastainen pätee, jos ohjelmointi nähdään työkaluna. Tämä näyttäisi viittaavan siihen, että Juuri 11 korostaa ohjelmointia omana kokonaisuutenaan, kun taas jotkin muut oppikirjat yleisesti ottaen suosivat näkemystä ohjelmoinnista matematiikassa käytettävänä työkaluna. Ei voida olla varmoja siitä, että vaikuttaako oppikirja siihen, miten opettajat suhtautuvat ohjelmointiin vai ohjaako opettajien suhtautuminen ohjelmointiin käyttämään tiettyjä oppimateriaaleja.



Kuva 13: Erilaisten oppimateriaalien suhteelliset osuudet ohjelmoinnin merkityksen mukaan.

5 Yhteenveto

Tämän tutkielman tavoitteena on ollut tutkia Suomen lukioissa järjestettyä MAA11 opintojaksoa, joka on ainoa ohjelmoinnin opetusta sisältävä opintojakso LOPS2019:ssä. Tehdyn kyselyn tulosten perusteella voidaan sanoa, että ohjelmoinnin opetus vaihtelee lukioittain verrattain paljon, mutta on myös joitain asioita, jotka ovat ainakin tähän kyselyyn osallistuneiden lukioiden perusteella samoja kaikkialla Suomessa. Vaikka LOPS2019 ei mainitse mitään yksittäistä ohjelmointikieltä, jolla ohjelmointia tulisi opettaa, ovat kaikki vastauksen antaneet lukiot silti päätyneet käyttämään Pythonia. Tämän lisäksi joissakin lukioissa käytetään taulukkolaskentasovelluksia, kuten Exceliä, tai matematiikka varten kehitettyjä ohjelmistoja, kuten GeoGebraa. Pythonin suosiolle saattaa olla monia tekijöitä. Esimerkiksi, koska aikaisemmassa koulutusvaiheessa käytetään Pythonia (katso esim. Järvenpää 2019), on nähty järkeväksi jatkaa saman kielen käyttöä. Suuri vaikutus voi olla myös sillä, että MAOL on opettajien tukimateriaalissa nimennyt juuri Pythonin mahdollisena opetuskielenä (Matemaattisten Aineiden Opettajien Liitto 2019). Muita yhteneväisyyksiä kaikkien vastauksen antaneiden lukioiden välillä ovat, että opintojaksolla opiskellaan ohjelmoinnin perusteista ehto- ja toistolauseiden käyttöä ja että oppilaat kirjoittavat ohjelmakoodia opintojakson aikana. Yleensä oppilaat kirjoittavat tietokoneohjelman itsenäisesti, mutta muutamissa lukioissa oppilaat ohjelmoivat ainoastaan opettajan johdolla. Voidaankin siis todeta, että vaikka ohjelmoinnin opetuksen toteutukseen ei oteta kantaa LOPS2019:ssä, niin käytännössä oppilaat kuitenkin oppivat ohjelmoimaan opintojaksolla.

Kaikkien muiden kysymysten vastauksissa eroja ilmenee joko jonkin verran tai hyvin paljon. Esimerkiksi ainoastaan yhdessä lukiossa on mainittu palikkaohjelmoinnin käyttö ohjelmoinnin opetuksessa, kun taas pari- tai ryhmätyöskentelyä käytetään lähes 75 prosentissa vastauksen antaneista lukioista. Tämän tutkielman kannalta mielenkiintoisia eroja on ollut ohjelmoinnin opetukseen käytettävä oppituntimäärä sekä opettajien suhtautuminen ohjelmointiin. Oppituntimäärien perusteella lähes kaikissa vastauksen antaneissa lukioissa käytetään vähintään neljä 75 minuutin oppituntia ohjelmoinnin opetukseen. Oppituntimäärä näyttää vaikuttavan useaan muuttujaan ja esimerkiksi mitä enemmän ohjelmointia opintojakso sisältää, sen enemmän esimerkiksi opetettavia teknologioita käydään läpi opintojakson aikana. Samoin ne opettajat, jotka sisällyttävät enemmän ohjelmointia käsitteleviä oppitunteja opintojaksoon, suhtautuvat myönteisemmin ohjelmointiin suhteessa matematiikkaan. Valtaosa kyselyyn vastanneista opettajista näkee ohjelmoinnin omana kokonaisuutena, joka on yhtä tärkeä kuin matematiikan kokonaisuus, mutta merkittävä vähemmistö pitää ohjelmointia vähemmän tärkeänä matematiikkaan verrattuna tai ainoastaan

matematiikassa käytettävänä työkaluna. Tulosten perusteella voidaan todeta, että myös Suomessa on havaittavissa Fuentes Martinezin kuvailemien duaaliopetuksen ja sirotellun opetuksen kaltaisia opetusstrategioita (Fuentes Martinez 2024). Opettajien suhtautumisella ohjelmointiin näyttäisi kyselyn tulosten perusteella olevan vaikutuksia siihen, ohjelmoivatko oppilaat itsenäisesti opintojakson aikana, sillä itsenäinen ohjelmointi oli paljon harvinaisempaa niissä lukioissa, joissa ohjelmointia pidettiin vähemmän tärkeänä kuin matematiikkaa.

5.1 Pohdinta ja ehdotuksia jatkotutkimukselle

Tämän tutkielman tuloksiin on suhtauduttava varauksella. Yleisesti ottaen tulokset, kuten että ympäri Suomen MAA11 opintojaksolla todella opetetaan ohjelmointia ja että oppilaille opetetaan ehto- ja toistolauseita, ovat melko luotettavia. Muiden tulosten kohdalla on oltava varovaisia sen suhteen, mitä johtopäätöksiä niistä tehdään. Esimerkiksi oppituntimäärät ovat tärkeitä ja keskiössä tässä tutkielmassa, mutta kyselylomakkeessa ei ole kysytty tarkkaan tietoa näistä oppitunneista. Sen sijaan oppituntimääriä on pyritty kartoittamaan antamalla muutamia välejä, joille oppituntien määrä sijoittuu. Tämä tarkoittaa sitä, että näiden välien sisällä saattaa olla suuria eroja eivätkä vastausryhmittymät todennäköisesti ole yhtä yhteneväisiä, kuin tässä tutkielmassa niitä on käsitelty. Erityisesti nyt jää epäselväksi, ovatko ne lukiot, joissa ohjelmoinnin opetukseen on varattu kuusi oppituntia ja täten kävisivät läpi kaikki Juuri 11 oppikirjassa esitellyt ohjelmointiin liittyvät kappaleet, todella erilaisia verrattuna niihin lukioihin, joissa oppitunteja on vähintään seitsemän. Oppituntimääriin liittyy myös muita kysymyksiä, kuten vaikuttaako oppituntien pituus ohjelmoinnin opetukseen, tai mikä on oppituntien hajonta yli seitsemän oppitunnin ryhmässä. Jatkotutkimuksessa olisikin hyvä ottaa oppituntimäärät paremmin huomioon.

Toinen merkittävä löytö tämän tutkielman perusteella on, että opettajien suhtautuminen ohjelmointiin vaikutti kysymysten vastauksiin. Esimerkiksi se, että itsenäinen ohjelmointi työskentelytapana näyttäisi riippuvan opettajan suhtautumisesta ohjelmointiin on mielenkiintoinen, mutta tämän tutkielman puitteissa asiaan ei päästä tämän syvemmälle. On hyvin luonnollisen kuuloista, että opettajat, jotka kokevat ohjelmoinnin tärkeänä, sisällyttävät enemmän ohjelmointia opintojaksoon kuin ne opettajat, jotka kokevat ohjelmoinnin vähemmän tärkeänä. Oiva jatkotutkimuksen kohde olisikin tutkia tarkemmin opettajien suhtautumista ohjelmoinnin opetukseen ja miten se näkyy käytännön opetustoteutuksessa. Erityisesti mielenkiintoista olisi tietää, määrääkö opettajan suhtautuminen ohjelmointiin oppituntien määrän, käytettävät oppimateriaalit ja opetusmetodit, vai onko niin, että pääsyynä

tälle ovat pikemminkin valitut oppimateriaalit tai vastaavasti aikataululliset syyt. Luonnollisesti tämä tutkielma on myös rajoittunut siinä, että vastauksia on saatu ainoastaan noin viidenneksestä Suomen lukioista. Täten ei voida olla täysin varmoja siitä, kuinka hyvin tämän tutkielman tulokset yleistyvät koko Suomen tasolle.

Lähdeluettelo

Alegre, Fernando et al. (2020). "Introduction to Computational Thinking: A New High School Curriculum using CodeWorld". Teoksessa: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. SIGCSE '20. New York, NY, USA: Association for Computing Machinery, s. 992–998. ISBN: 978-1-4503-6793-6. DOI: 10.1145/3328778.3366960. URL: https://dl.acm.org/doi/10.1145/3328778.3366960 (viitattu 02. 04. 2024).

Balanskat, Anja ja Katja Engelhardt (2015). *Computing our Future*. DOI: 10.13140/RG.2.1.5029.9048.

Barr, Valerie ja Chris Stephenson (2011). "Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?" *ACM Inroads* 2. DOI: 10.1145/1929887.1929905.

Bau, David et al. (2017). "Learnable programming: blocks and beyond". *Communications of the ACM* 60.6, s. 72–80. ISSN: 0001-0782. DOI: 10.1145/3015455. URL: https://dl.acm.org/doi/10.1145/3015455 (viitattu 02. 04. 2024).

Belmar, Héctor (2022). "Review on the teaching of programming and computational thinking in the world". *Frontiers in Computer Science* 4. ISSN: 2624-9898. URL: https://www.frontiersin.org/articles/10.3389/fcomp.2022.997222 (viitattu 02. 04. 2024).

Bocconi, Stefania et al. (2022). *Reviewing Computational Thinking in Compulsory Education*. en. Publication Title: JRC Publications Repository. DOI: 10.2760/126955. URL: https://publications.jrc.ec.europa.eu/repository/handle/JRC128347 (viitattu 02. 04. 2024).

Brown, Neil C. C. ja Greg Wilson (2018). "Ten quick tips for teaching programming". en. *PLOS Computational Biology* 14.4. Publisher: Public Library of Science, e1006023. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1006023. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1006023 (viitattu 09. 04. 2024).

Çınar, Murat ja Hakan Tüzün (2021). "Comparison of object-oriented and robot programming activities: The effects of programming modality on student achievement, abstraction, problem solving, and motivation". en. *Journal of Computer Assisted Learning* 37.2, s. 370–386. ISSN: 1365-2729. DOI: 10.1111/jcal.12495. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/jcal.12495 (viitattu 02. 04. 2024).

Cohen, Jacob (1988). *Statistical Power Analysis for the Behavioral Sciences*. English. 2nd edition. Hillsdale, N.J: Routledge. ISBN: 978-0-8058-0283-2.

CSTA (2024). en-us. URL: http://www.csteachers.org (viitattu 02. 04. 2024).

Dagiene, Valentina, Juraj Hromkovic ja Regula Lacher (2021). "Designing informatics curriculum for K-12 education: From Concepts to Implementations". en. *Informatics in Education*. ISSN: 1648-5831, 2335-8971. DOI: 10.15388/infedu.2021.22. URL: https://infedu.vu.lt/doi/10.15388/infedu.2021.22 (viitattu 02. 04. 2024).

Fisler, Kathi et al. (2021). "Evolving a K-12 Curriculum for Integrating Computer Science into Mathematics". Teoksessa: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. SIGCSE '21. New York, NY, USA: Association for Computing Machinery, s. 59–65. ISBN: 978-1-4503-8062-1. DOI: 10.1145/3408877.3432546. URL: https://dl.acm.org/doi/10.1145/3408877.3432546 (viitattu 02. 04. 2024).

Fuentes Martinez, Ana (2024). "Practice beyond technology when programming and mathematics teaching converge". eng. URL: https://urn.kb.se/resolve?urn=urn:nbn:se:hv:diva-21050 (viitattu 02. 04. 2024).

Furber, Steve (2012). *Shut down or restart? The way forward for computing in UK schools*. Tekninen raportti. London: The Royal Society. URL: https://royalsociety.org/topics-policy/projects/computing-in-schools/report/ (viitattu 02. 04. 2024).

Fülöp, Melinda Timea et al. (2022). "Development of Computational Thinking Using Microcontrollers Integrated into OOP (Object-Oriented Programming)". en. *Sustainability* 14.12, s. 7218. ISSN: 2071-1050. DOI: 10.3390/su14127218. URL: https://www.mdpi.com/2071-1050/14/12/7218 (viitattu 02. 04. 2024).

Gal-Ezer, Judith ja David Harel (1998). "What (else) should CS educators know?" *Communications of the ACM* 41.9, s. 77–84. ISSN: 0001-0782. DOI: 10.1145/285070.285085. URL: https://doi.org/10.1145/285070.285085 (viitattu 02. 04. 2024).

Harris, Paul A., Robert Taylor, Brenda L. Minor et al. (2019). "The REDCap consortium: Building an international community of software platform partners". *Journal of Biomedical Informatics* 95, s. 103208. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2019.103208. URL: https://www.sciencedirect.com/science/article/pii/S1532046419301261 (viitattu 13. 05. 2024).

Harris, Paul A., Robert Taylor, Robert Thielke et al. (2009). "Research electronic data capture (REDCap)—A metadata-driven methodology and workflow process for providing translational research informatics support". *Journal of Biomedical Informatics* 42.2, s. 377–381. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2008.08.010. URL: https://www.sciencedirect.com/science/article/pii/S1532046408001226 (viitattu 13. 05. 2024).

Harsunkorpi, Jukka et al. (2023). *Moodi MAA 11*. Finnish. 2nd. Helsinki: Sanoma Pro Oy. ISBN: 978-952-63-5850-5. URL: https://tuotteet.sanomapro.fi/bu595582-moodi-pitka-maa11-lops21.html.

Hubwieser, Peter et al. (2011). "Computer science/informatics in secondary education". en. Teoksessa: *Proceedings of the 16th annual conference reports on Innovation and technology in computer science education - working group reports*. Darmstadt Germany: ACM, s. 19–38. ISBN: 978-1-4503-1122-9. DOI: 10.1145/2078856.2078859. URL: https://dl.acm.org/doi/10.1145/2078856.2078859 (viitattu 02. 04. 2024).

Hähköniemi, Markus et al. (2022). *Juuri 11 — MAA11 Algoritmit ja lukuteoria*. Finnish. 2nd. Helsinki: Otava. ISBN: 978-951-1-36302-6. URL: https://otava.kauppakv.fi/sivu/tuote/juuri-11-lops21-/2834143.

Järvenpää, Jouko (2019). *Python-ohjelmoinnin opas opettajalle | Edita*. Finnish. Helsinki: Edita. ISBN: 978-951-37-7466-0. URL: https://shop.edita.fi/oppiminen/tuote/python-ohjelmoinnin-opas-opettajalle (viitattu 24. 04. 2024).

Kaila, E. et al. (2018). "Technology-enhanced programming courses for upper secondary school students". Teoksessa: *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, s. 0683–0688. DOI: 10.23919/MIPRO.2018.8400128.

Kaufmann, Odd Tore ja Børre Stenseth (2021). "Programming in mathematics education". *International Journal of Mathematical Education in Science and Technology* 52.7, s. 1029–1048. ISSN: 0020-739X. DOI: 10.1080/0020739X.2020.1736349. URL: https://doi.org/10.1080/0020739X.2020.1736349 (viitattu 02. 04. 2024).

Lv, Lin, Baichang Zhong ja Xiaofan Liu (2023). "A literature review on the empirical studies of the integration of mathematics and computational thinking". en. *Education and Information Technologies* 28.7, s. 8171–8193. ISSN: 1573-7608. DOI: 10.1007/s10639-022-11518-2. URL: https://doi.org/10.1007/s10639-022-11518-2 (viitattu 02. 04. 2024).

Mannila, Linda, Mia Peltomäki ja Tapio Salakoski (2006). "What about a simple language? Analyzing the difficulties in learning to program". *Computer Science Education* 16.3, s. 211–227. ISSN: 0899-3408. DOI: 10.1080/08993400600912384. URL: https://doi.org/10.1080/08993400600912384 (viitattu 02. 04. 2024).

Matemaattisten Aineiden Opettajien Liitto (2019). *Matematiikan Lops-tukimateriaalit*. fi. URL: https://www.oph.fi/fi/koulutus-ja-tutkinnot/matematiikan-lops-tukimateriaalit (viitattu 02. 04. 2024).

Merino-Armero, José Miguel, José Antonio González-Calero ja Ramón Cózar-Gutiérrez (2022). "Computational thinking in K-12 education. An insight through meta-analysis". *Journal of Research on Technology in Education* 54.3, s. 410–437. ISSN: 1539-1523. DOI: 10.1080/15391523.2020.1870250. URL: https://doi.org/10.1080/15391523.2020.1870250 (viitattu 02. 04. 2024).

Muhammad, Ilham et al. (2024). "Computational Thinking Research in Mathematics Learning in the Last Decade: A Bibliometric Review". en. *International Journal of Education in Mathematics, Science and Technology* 12.1, s. 178–202. URL: https://eric.ed.gov/?id=EJ1408639 (viitattu 02. 04. 2024).

Nojonen, Tapio (2024). "Ohjelmoinnin opetus lukioiden paikallisissa opetussuunnitelmissa". fin. URL: https://www.utupub.fi/handle/10024/176665 (viitattu 02. 04. 2024).

Nousiainen, Saara ja Anne Kivistö (2022). "Ohjelmoinnin opetuksen arviointi lukiokoulutuksessa". fi. URL: https://karvi.fi/publication/ohjelmoinnin-opetuksen-arviointi-lukiokoulutuksessa/.

Opetushallitus (2014). *Perusopetuksen opetussuunnitelman perusteet*. fi. Publication Title: Opetushallitus. URL: https://www.oph.fi/fi/koulutus-ja-tutkinnot/perusopetuksen-opetussuunnitelman-perusteet (viitattu 02. 04. 2024).

— (2019). *Lukion opetussuunnitelmien perusteet*. fi. Publication Title: Opetushallitus. URL: https://www.oph.fi/fi/koulutus-ja-tutkinnot/lukion-opetussuunnitelmien-perusteet (viitattu 02. 04. 2024).

Ou, Qizhong et al. (2023). "Investigation and analysis of the current situation of programming education in primary and secondary schools". en. *Heliyon* 9.4, e15530. ISSN: 2405-8440. DOI: 10.1016/j.heliyon.2023.e15530. URL: https://www.sciencedirect.com/science/article/pii/S2405844023027378 (viitattu 02. 04. 2024).

Papert, Seymour (1980). *Mindstorms*. English. New York, NY, USA: Basic Books. ISBN: 978-1-5416-7512-4.

Robins, Anthony, Janet Rountree ja Nathan Rountree (2003). "Learning and Teaching Programming: A Review and Discussion". *Computer Science Education* 13.2, s. 137–172. ISSN: 0899-3408. DOI: 10.1076/csed.13.2.137.14200. URL: https://doi.org/10.1076/csed.13.2.137.14200 (viitattu 02. 04. 2024).

Saidova, Dilfuza Ergashovna (2022). "Analysis of the Problems of the Teaching Object-Oriented Programming to Students". en. *International Journal of Social Science Research and Review* 5.6, s. 229–234. ISSN: 2700-2497. DOI: 10.47814/ijssrr.v5i6.418. URL: https://ijssrr.com/journal/article/view/418 (viitattu 02. 04. 2024).

Scherer, Ronny, Fazilat Siddiq ja Bárbara Sánchez Viveros (2020). "A meta-analysis of teaching and learning computer programming: Effective instructional approaches and conditions". *Computers in Human Behavior* 109, s. 106349. ISSN: 0747-5632. DOI: 10.1016/j.chb.2020.106349. URL: https://www.sciencedirect.com/science/article/pii/S0747563220301023 (viitattu 08. 04. 2024).

Seehorn, Deborah et al. (2017). "The CSTA Standards Task Force". en. URL: https://www.csteachers.org/Page/standards (viitattu 02. 04. 2024).

Sim, Tze Ying ja Sian Lun Lau (2022). "Review on Challenges and Solutions in Novice Programming Education". Teoksessa: *2022 IEEE International Conference on Computing (ICOCO)*, s. 55–61. DOI: 10.1109/ICOCO56118.2022.10031657.

Szabo, Claudia et al. (2019). "Fifteen Years of Introductory Programming in Schools: A Global Overview of K-12 Initiatives". Teoksessa: *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*. Koli Calling '19. New York, NY, USA: Association for Computing Machinery, s. 1–9. ISBN: 978-1-4503-7715-7. DOI: 10.1145/3364510.3364513. URL: https://doi.org/10.1145/3364510.3364513 (viitattu 02. 04. 2024).

Webb, Mary et al. (2017). "Computer science in K-12 school curricula of the 2lst century: Why, what and when?" en. *Education and Information Technologies* 22.2, s. 445–468. ISSN: 1573-7608. DOI: 10.1007/s10639-016-9493-x. URL: https://doi.org/10.1007/s10639-016-9493-x (viitattu 02. 04. 2024).

Weintrop, David, Elham Beheshti et al. (2016). "Defining Computational Thinking for Mathematics and Science Classrooms". en. *Journal of Science Education and Technology* 25.1, s. 127–147. ISSN: 1573-1839. DOI: 10.1007/s10956-015-9581-5. URL: https://doi.org/10.1007/s10956-015-9581-5 (viitattu 11. 04. 2024).

Weintrop, David ja Uri Wilensky (2019). "Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms". *Computers & Education* 142, s. 103646. ISSN: 0360-1315. DOI: 10.1016/j.compedu.2019.103646. URL: https://www.sciencedirect.com/science/article/pii/S036013151930199X (viitattu 02. 04. 2024).

Wing, Jeannette M. (2006). "Computational thinking". *Communications of the ACM* 49.3, s. 33–35. ISSN: 0001-0782. DOI: 10.1145/1118178.1118215. URL: https://dl.acm.org/doi/10.1145/1118178.1118215 (viitattu 02. 04. 2024).

Ye, Huiyan et al. (2023). "Integration of computational thinking in K-12 mathematics education: a systematic review on CT-based mathematics instruction and student learning". en. *International Journal of STEM Education* 10.1, s. 3. ISSN:

2196-7822. DOI: 10.1186/s40594-023-00396-w. URL: https://doi.org/10.1186/s40594-023-00396-w (viitattu 02. 04. 2024).

Ylioppilastutkintolautakunta (2024). URL: https://www.ylioppilastutkinto.fi/fi.

Liitteet

A Kyselylomake

Ohjelmoinnin opetus MAA11 opintojaksolla

Hei ja kiitos mielenkiinnostasi! Tuet arvokasta tutkimusta ohjelmoinnin opetuksesta Suomen lukioissa, kun vastaat alla oleviin kysymyksiin.

Kysely alkaa perustietojen keräämisellä ja etenee tämän jälkeen varsinaisesti ohjelmoinnin opetusta koskeviin aihealueisiin. Kyselyn täyttäminen vie arviolta noin 2-3 minuuttia.

1)	Lukion nimi	_____
2)	Kunnan nimi	_____ (Sen kunnan nimi, jossa lukio sijaitsee. Käytetään mahdollisten alueellisten samankaltaisuuksien havaitsemiseen)
3)	Kuinka monta 75 minuutin oppituntia opintojaksolla käytetään ohjelmointiin?	☐ 0 ☐ 1-3 ☐ 4-6 ☐ 7+ (Suurin piirtein kuinka monta oppituntia toteutettu siten, että oppilaat tuottavat itse ohjelmakoodia. Jos toteutus 45 minuutin oppitunteina, niin voit pyöristää siten, että kolme 45 minuutin oppituntia vastaa kahta 75 minuutin oppituntia)

Seuraavassa osiossa kysytään kysymyksiä opintojaksolla käytettävistä ohjelmointikielistä ja -tekniikoista

Mikäli MAA11 tai sitä vastaavaa opintojaksoa ei ole toteutunut tai toteutuksessa ei ollut ohjelmoinnin opetusta, voit vastata loppuihin kysymyksiin sen perusteella, miten olet suunnitellut MAA11 opintojakson

4)	Mitä ohjelmointikieltä tai -kieliä opintojaksolla käytetään?	☐ Python ☐ Scratch ☐ Java ☐ C# ☐ MatLab ☐ Muu, mikä/mitä: _____ (Esimerkiksi Excel ja muut taulukkoslaskentaohjelmistot voi merkitä kohtaan "Muu, mikä/mitä")
5)	Mitä ohjelmointitekniikoita oppilaat käyttävät opintojaksolla?	☐ Ehtolause (if-else) ☐ Toistolause (while, for) ☐ Olio-ohjelmointi (omien luokkien luonti) ☐ Kirjastojen lisäys (esim. import-komennot) ☐ Funktio (ohjelmakoodiin kirjoitettu parametrisoitu funktio) ☐ Tietotyyppien muunnos (esim. string-muuttuja int-muuttujaksi) ☐ Muu, mikä/mitä: _____

Seuraavassa osiossa kysytään kysymyksiä siitä, miten ohjelmointia opetetaan opintojaksolla

6) Mikä näistä väittämistä parhaiten kuvastaa ohjelmoinnin opetusta opintojaksolla?

- ◯ Ohjelmointia opetetaan omana kokonaisuutenaan, joka on yhtä tärkeä kuin matematiikan kokonaisuus
- ◯ Ohjelmointia opetetaan omana kokonaisuutenaan, joka on vähemmän tärkeä kuin matematiikan kokonaisuus
- ◯ Ohjelmointia opetetaan omana kokonaisuutenaan, mutta vain, jos siihen riittää aikaa
- ◯ Ohjelmointia opetetaan matematiikassa käytettävänä työkaluna, jota käytetään usein
- ◯ Ohjelmointia opetetaan matematiikassa käytettävänä työkaluna, jota käytetään toisinaan
- ◯ Ohjelmointia opetetaan matematiikassa käytettävänä työkaluna, jota käytetään vain, jos siihen jää aikaa
- ◯ Ohjelmoinnista puhutaan, mutta oppilaita ei opeteta ohjelmoimaan

7) Miten ohjelmoinnin opetusta toteutetaan opintojaksolla?

- ☐ Oppilaat seuraavat opettajan esittämää esimerkkiä, mutta eivät ohjelmoi itsenäisesti
- ☐ Oppilaat tutkivat opettajan jakaman ohjelman toimintaa (esim. muuttujien arvojen vaihtaminen)
- ☐ Oppilaat muokkaavat opettajan jakamaa ohjelmaa (esim. uuden funktion lisääminen ohjelmaan)
- ☐ Oppilaat luovat oman ohjelman opettajan johdolla (esim. opettaja näyttää taululta kaikki välivaiheet ja oppilaat seuraavat perässä)
- ☐ Oppilaat luovat oman ohjelman itsenäisesti

8) Mitä seuraavista opetustekniikoista opintojaksolla käytetään ohjelmoinnin opetuksessa?

- ☐ Pari- ja/tai ryhmätyöskentely
- ☐ Robotiikka
- ☐ Opetuspelit
- ☐ Videotutoriaalit/opetusvideot
- ☐ Ohjelmointigrafiikat (esimerkiksi turtle-grafiikat)
- ☐ Palikkaohjelmointi

9) Mitä oppimateriaaleja opintojaksolla käytetään?

- ☐ Juuri11 oppikirja (Otava)
- ☐ Moodi11 oppikirja (SanomaPro)
- ☐ Jonkin muun kustantajan oppikirja
- ☐ Itse tehty opetusmateriaali
- ☐ Kollegoiden tai muiden tahojen jakama opetusmateriaali