Tampereen yliopisto

Else Pasanen

Nelikulmioiden oppiminen ohjelmoinnin avulla

Informaatioteknologian ja viestinnän tiedekunta
Pro gradu -tutkielma
Matematiikka
Joulukuu 2019

TIIVISTELMÄ

Else Pasanen: Nelikulmioiden hierarkia alakoulussa
Pro gradu -tutkielma
Tampereen yliopisto
Matematiikan ja tilastotieteen tutkinto-ohjelma
Joulukuu 2019

Tutkimuksessa tarkasteltiin matematiikan oppimista ohjelmoinnin avulla alakoulussa. Opetuskokeilussa laadittiin oppilaille oppimateriaalia, joiden avulla he opiskelivat nelikulmioiden hierarkiaa visuaalista ohjelmointiympäristöä apunaan käyttäen. Tutkimuksessa tarkasteltiin oppimistuloksia sekä oppilaiden asenteita ja mielikuvia ohjelmoinnista osana matematiikan opiskelua.

Tutkimukseen osallistui kaksi luokkaa kahdesta eri koulusta ja se toteutettiin kehittämistutkimuksena. Luokissa oli yhteensä 25 oppilasta, jotka osallistuivat tutkimukseen. Tutkimuksen tavoitteena oli tutkia, miten matematiikan ja ohjelmoinnin opetus voidaan yhdistää mielekkäästi alakoulun matematiikan opetuksessa.

Tutkimustulosten perusteella opetuksen jälkeen oppilaat osasivat nimetä nelikulmioiden luokkia sekä nimetä niiden erottavia ja yhdistäviä tekijöitä melko hyvin. Sen sijaan nelikulmioiden välisten hierarkiasuhteiden ymmärtäminen tuotti oppilaille vaikeuksia. Tutkimustulokset jakautuivat selvästi luokkien välillä, kun oppilailta kysyttiin heidän asenteistaan ohjelmointia kohtaan. Toinen luokka koki ohjelmoinnin mielenkiintoisena lisänä matematiikan opiskeluun ja pitivät sitä tärkeänä taitona. Toinen luokka taas suhtautui ohjelmointiin huomattavasti skeptisemmin.

Ohjelmoinnin opetuksesta peruskoulussa tarvittaisiin lisätutkimusta. Tutkimustuloksissa oli havaittavissa suuria eroja oppilaiden asenteissa luokkien välillä, mutta tutkimus ei antanut vastauksia erojen syistä. Oppilailla oli myös selkeästi vaikeuksia hahmottaa nelikulmioiden hierarkiaa. Olisi tärkeää tutkia ovatko alakoululaiset liian nuoria ymmärtämään abstrakteja käsitteitä kuten hierarkia. Lisäksi ohjelmointi on kytketty peruskoulussa vahvasti osaksi matematiikan opetusta, mutta oppimateriaalia ja tutkimusta siitä, miten ohjelmoinnilla voidaan tukea erityisesti matematiikan oppimista, on erittäin vähän.

Avainsanat: Geometria, alakoulu, ohjelmointi, kehittämistutkimus
Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -ohjelmalla.

Sisältö

1 Johdanto

Tietotekniikka on tällä hetkellä osa ihmisten arkea. Tietotekniikan arkipäiväistymien on huomioitu myös viimeisimmässä opetussuunnitemassa, jossa ohjelmointitaito lisättiin opetussuunnitelman tavoitteisiin. Ohjelmointi on huomioitu erityisesti osana matematiikan opetusta opetussuunnitelmassa, vaikka se on mainittu myös osana muiden oppiaineiden sisältöä. [21] Ohjelmoinnin opetuksesta osana matematiikan opetusta on kuitenkin olemassa erittäin vähän tutkimusta. Tässä tutkimuksessa tarkastellaankin, voidaanko ohjelmoinnin keinoin syventää oppilaan matemaattista ymmärrystä.

Useat eri yritykset ja yliopistot ovat osallistuneet viime aikoina erilaisten lapsille ja nuorille suunnattujen graafisten ohjelmointikielien kehitykseen. Tarkoituksena on saada oppilaat innostumaan ohjelmoinnista ja saada tulevaisuudessa enemmän opiskelijoita ja sitä kautta ammattilaisia ohjelmistoalalle.

Tutkimuksen alussa tehdään katsaus aikaisempiin tutkimuksiin ja aihepiiriin liittyvään kirjallisuuteen. Aluksi luodaan katsaus matematiikan oppimista käsittelevään kirjallisuuteen. Tässä luvussa käsitellään matematiikan oppimisen eri teorioita ja oppimisprosessin vaiheita. Luvussa 3 tarkastellaan peruskoulun opetussuunnitelmaa tutkimuksen aiheen valossa. Tämän jälkeen luodaan katsaus ohjelmointiin ja algoritmiseen ajatteluun luvussa 4 sekä nelikulmioihin ja niiden hierarkiaan luvussa 5. Kirjallisuuskatsauksen lopuksi käsitellään vielä kehittämistutkimusta.

Tutkimuksen tavoitteena on tutkia, voidaanko ohjelmoinnin keinoin syventää oppilaiden matemaattista ajattelua. Lisäksi tutkitaan oppilaiden suhtautumista ohjelmointiin osana matematiikan opiskelua. Tutkimuksessa kehitettiin tehtäviä, joiden avulla oppilaat opettelivat ohjelmoinnin alkeita. Ohjelmointitehtävin aiheina oli erilaisia nelikulmioihin ja niiden hierarkioihin liittyviä tehtäviä. Tehtävissä oli mukana myös kielentämistehtäviä, joissa oppilaiden piti selittää, miten he päätyivät tekemäänsä ratkaisuun. Tehtävin tekemisen jälkeen oppilaat täyttivät kyselylomakkeen, jossa kysyttiin oppilaiden asenteisiin ja tehtävien mielekkyyteen liittyviä kysymyksiä. Lisäksi osaa oppilaista haastateltiin. Haastatteluilla pyrittiin selvittämään oppilaiden ymmärrystä nelikulmioiden hierarkiasta. Lopuksi pohditaan tutkimuksen tuloksia ja niiden herättämiä kysymyksiä, joiden pohjalta olisi tarpeen tehdä lisätutkimusta.

2 Matematiikan oppiminen ja osaaminen

Tässä luvussa käsitellään matemaattisen opetuksen piirteitä sekä tutkimuksen kannalta keskeisimpiä matematiikan oppimiseen ja osaamiseen liittyviä teorioita. Matemaattisen osaamisen piirteet luovat pohjan matematiikan oppimiselle. Tämän jälkeen tarkastellaan ongelmanratkaisua ja siihen tarvittavaa osaamista sekä tutkivaa oppimista oppimisstrategiana. Lopuksi käsitellään kielentämistä osana matematiikan opiskelua ja matemaattisen osaamisen kehittymistä.

Matematiikan oppiminen ja osaaminen koostuvat monesta eri osa-alueesta. Matematiikan osaaminen koostuu pääpiirteittäin matematiikan teorian hallitsemisesta, päättelystä ja yrittelijäisyydestä. Koulussa matematiikan opetuksen tarkoituksena on pitkälti tukea näiden piirteiden kehittymistä. Ongelmanratkaisutaito on keskeisessä asemassa matematiikan osaamisessa. Matematiikan osaamisen piirteitä käsitellään luvussa 2.1 ja ongelmanratkaisua luvussa 2.2. Matemaattinen ongelmanratkaisu on erityisen keskeisessä asemassa tutkivan oppimisen teoriassa, johon perehdytään luvussa 2.3. Tässä tutkimuksessa matematiikan opettamista on lähestytty tutkivan oppimisen näkökulmasta. Luvun lopussa käsitellään kielentämistä, joka tukee kaikkia edellä mainittuja matematiikan oppimisen osa-alueita. Matematiikan kielen hallitseminen on erityisen tärkeää matematiikan ymmärtämisen ja matematiikkaan liittyvän tiedon kannalta.

2.1 Matemaattisen osaamisen piirteet

Matemaattinen osaaminen voidaan katsoa koostuvan viidestä piirteestä, jotka ovat käsitteellinen ymmärtäminen (*conceptual understanding*), proseduraalinen sujuvuus (*procedural fluency*), strateginen kompetenssi (*strategic competence*), mukautuva päättely (*adaptive reasoning*) ja yrittelijäisyys (*productive disposition*) [15, s. 96]. Kaikki nämä piirteet ovat toisistaan riippuvia ja jokaiselta oppilaalta löytyy jossain määrin kaikkia piirteitä. Oppilaan matemaattinen osaaminen koostuu näistä piirteistä ja etenkin alakoulussa matematiikan opetuksen tulisi tukea näiden piirteiden kehittymistä.

Käsitteellinen ymmärtäminen tarkoittaa käsitteiden ymmärrystä ja niiden liittämistä hyvin organisoituun tietorakenteeseen. Vaikka oppilaan käsitteellinen ymmärrys olisi vahva, hän ei välttämättä osaa tuoda ymmärrystään verbaalisesti esille. Matematiikan opetuksessa käsitteellisen ymmärtämisen kehittymistä tuetaan matematiikan teorian opetuksella. Oppilaat opettelevat nimeämään erilaisia matemaattisia ilmiöitä ja rakentavat yhteyksiä eri termien välille. Esimerkiksi oppilas oppii nimeämään tietynlaisen geometrisen kuvion nelikulmioksi ja myöhemmin suorakulmioksi. Oppilas, joka on taitava käsitteiden ymmärtämisessä, huomaa, että kyseinen kuvio on sekä nelikulmio että suorakulmio. Lisäksi kaikki suorakulmiot ovat väistämättä nelikulmioita, mutta nelikulmiot eivät välttämättä ole suorakulmioita. Tässä tutkimuksessa keskitytään erityisesti käsitteellisen ymmärtämisen tarkasteluun. Tutkimuksessa pyritään tarkastelemaan alakoululaisten ymmärrystä nelikulmioiden

hierarkiasta. Hierarkian ymmärtämisessä korostuu erityisesti siihen liittyvät käsitteet ja niiden strukturointi.

Proseduraalinen sujuvuus tarkoittaa laskentataidon hallitsemista. Oppilas tuntee proseduurit, tietää missä niitä käytetään ja osaa käyttää niitä tarkoituksenmukaisesti. Laskeminen voi olla sujuvaa vain, jos oppilas hallitsee matemaattiset proseduurit täsmällisesti, tehokkaasti ja tarkoituksenmukaisesti. Alakoulun matematiikassa proseduraalinen sujuvuus tarkoittaa etenkin teknistä laskemista ja sen sujuvuutta. Perinteisesti matematiikan opetuksessa on keskitytty erityisesti proseduraalisen sujuvuuden kehittymiseen ja sitä on testattu etenkin paljon laskentaa sisältävillä kokeilla. Tässä tutkimuksessa keskitytään muiden matemaattisen osaamisen piirteiden tarkasteluun.

Strategista kompetenssia omaava oppilas tuntee ja hallitsee useita ongelmanratkaisustrategioita. Oppilas osaa muotoilla ongelman sellaiseen muotoon, että se on mahdollista ratkaista. Tällaista ongelmanratkaisukykyä varten oppilaan tulee hallita käsitteitä ja proseduureja, joita oppilas voi soveltaa ongelman muotoilussa ja ratkaisussa. Käsitteellinen ymmärtäminen ja proseduraalinen sujuvuus luovat pohjan strategiselle kompetenssille. Niiden avulla oppilas voi muotoilla ongelman sellaiseen muotoon, että hän kykenee sen ratkaisemaan. Alakoulun matematiikassa strategista kompetenssia pyritään kehittämään etenkin sanallisten tehtävien muodossa.

Mukautuva päättely on oppilaan kykyä ajatella käsitteiden ja tilanteiden välisiä suhteita. Tällainen looginen ajattelu vaatii oppilaalta riittävää tietopohjaa. Lisäksi tehtävän tulee olla oppilaan mielestä ymmärrettävä ja motivoiva sekä tehtävän kontekstin miellyttävä. Jos oppilas kokee tehtävän epämiellyttävänä, sen ratkaiseminen mukautuvan päättelyn avulla on haastavaa. Mukautuvassa päättelyssä olennaista on, että oppilas osaa perustella oman toimintansa. Mukautuva päättely on myös vahvasti sidoksissa strategiseen kompetenssiin ja konseptuaaliseen tietoon, sillä oppilas tarvitsee mukautuvaa päättelyä päättäessään strategiaa, jonka avulla hän ratkaisee tehtävän. Mukautuva päättely korostuu erityisesti ongelmanratkaisutilanteessa, jossa oppilaan pitää kyetä perustelemaan tekemänsä ratkaisu joillain tavalla. Alakoulussa oppilailta ei vaadita vielä vahvaa mukautuvan päättelyn taitoa. Sen sijaan etenkin nopeimmille laskijoille tarjotaan usein erilaisia ongelmatehtäviä, joiden ratkaisemiseen tarvitaan erityisesti tätä taitoa.

Oppilaan yritteliäisyys on oppilaan käsitystä itsestä oppijana ja matematiikasta oppiaineena. Yritteliäs oppilas näkee matematiikan opiskelun tärkeänä ja mielekkäänä. Lisäksi oppilas uskoo, että matematiikan ahkera opiskelu kannattaa ja hän uskoo omiin kykyihinsä ja mahdollisuuksiinsa oppia. Kaiken kaikkiaan yritteliäällä oppilaalla on positiivinen kuva matematiikasta yleisesti ja itsestään matematiikan opiskelijana. Positiivinen asenne edistää muiden piirteiden kehittymistä ja sillä on vaikutusta metakognition kautta oppilaan suorituksiin. Etenkin viimeisimmän opetussuunnitelman myötä matematiikan opetuksessa on pyritty huomioimaan se, että oppilaat saisivat positiivisia kokemuksia matematiikan oppimisesta ja siten motivoituisivat matematiikan osaamisen kehittämiseen. [15, s. 96-99]

2.2 Ongelmanratkaisu ja ongelmakentät

Ongelmanratkaisua on pidetty matemaattisen ajattelun ytimenä hyvin pitkään [15, s. 59]. Lisäksi uusin opetussuunnitelma mainitsee ongelmanratkaisutaitojen kehittämisen yhdeksi matematiikan opetuksen tavoitteeksi [21, s. 235]. Ongelmanratkaisutehtävät voidaan jaotella tehtäviin, joissa on määritelty alku- ja lopputila, toinen niistä tai sekä alku- että lopputila puuttuvat.

Berry [3, s. 51] jakaa ongelmanratkaisun koulumatematiikassa kolmeen osaan; perusongelmiin, tutkimustehtäviin ja mallintamiseen. Kahdessa viimeisessä osassa oppilas tarvitsee ratkaisuprosesseja, joita Polya [23] kuvaa seuraavasti. Prosessissa on neljä vaihetta, jotka ovat ongelman ymmärtäminen, suunnitelman laatiminen, sen toteutus ja tuloksen tarkistaminen. Polyan ongelmanratkaisuprosessi tulee parhaiten esille ongelmissa, joissa on alku- tai lopputila tai ei kumpaakaan. Tässä tutkimuksessa keskitytään ongelmiin, joissa alkutila on annettu. Toinen tapa hahmottaa ongelmanratkaisua on jakaa se osatekijöihin. Schoenfeldin [25, s. 348] mukaan näitä osatekijöitä ovat resurssit, strategiat, kontrolli ja emootio. Resurssit kuvaavat oppilaan kaikkea käytössä olevaa tietotaitoa. Kontrolli ja emootio tulevat esille metakognitioiden kautta. Strategioiden hallinta, tunnistaminen ja valitseminen korostuvat erityisesti ongelmanratkaisuissa. Metakognitiot ohjaavat erilaisten strategioiden valitsemisessa ja käytössä, joten metakognitiolla on keskeinen rooli ongelmanratkaisussa.

Dreyfusin ja Eisenbergin [7, s. 253-281] mukaan ongelmanratkaisussa kehittyneessä matemaattisessa ajattelussa tärkeitä ajattelun taitoja ovat esteettisyys, analoginen päättely, struktuurien ymmärtäminen, esittäminen, visuaalinen päättely ja käänteinen ajattelu. Näiden taitojen avulla ongelmanratkaisija pystyy näkemään ongelman useammasta näkökulmasta.

Esteettinen ajattelu ohjaa oppilasta lyhyisiin ja elegantteihin ratkaisuihin. Analoginen päättely on yhtenevien piirteiden löytämistä erityyppisten ongelmien välillä. Tämä auttaa uusien ongelmien ratkaisussa on uusien käsitteiden oppimisessa. Struktuurien ymmärtäminen on kykyä hahmottaa faktojen välisiä suhteita [15, s. 59-62]. Analogista päättelyä ja struktuurista ymmärtämistä voidaan kuvata yleisellä tasolla myös ongelmaan liittyvien termien selventämisenä ja selityksen rakentamisena. Nämä ovat oppilaan aikaisemmin oppimia asioita, joita hän voi käyttää uusien ongelmien ratkaisussa apuna [12, s. 217]. Ongelman struktuurin löytäminen auttaa oppilasta hahmottamaan mahdolliset operaatiot ongelmanratkaisussa ja se mahdollistaa analogian käyttämisen ratkaisuprosessissa. Lisäksi se tekee tiedon käsittelystä tehokkaampaa. Alakoulun matematiikassa korostuu analoginen päättely ja ongelman struktuurin löytäminen. Alakoulun matematiikassa ei vielä vaadita ratkaisuilta erityistä esteettisyyttä, mutta esteettisyys tarjoaa haasteita etenkin eriytettäessä ylöspäin.

Visuaalinen päättely on kuvien kautta ongelman ja sen ratkaisun hahmottamista. Ongelman esittäminen eri muodoissa auttaa usein ratkaisun keksimisessä. Liika visualisointi saattaa joissain tilanteissa yksinkertaistaa ongelmaa liikaa etenkin niiden oppilaiden kohdalla, jotka ovat edenneet pitkälle matematiikan opinnoissa. Sen sijaan alakoulussa ongelmat ovat pääsääntöisesti niin yksinkertaisia, että visualisointi ei

aiheuta ongelman liikaan yksinkertaistumista. Käänteisessä ajattelussa ongelmalla on jo olemassa ratkaisu, mutta oppilaan tehtävänä on päätellä ongelman alkutila ja ratkaisuprosessi [15, s. 59-62].

2.3 Tutkiva oppiminen

Matematiikassa ongelmanratkaisuun ja matematiikan soveltamiseen tarvitaan luovuutta eli kykyä keksiä uudenlaisia ratkaisuja ja ajatella omalla tavallaan. Luovassa prosessissa on neljä vaihetta, jotka ovat valmistelujakso, kypsymisjakso, ratkaisun löytyminen ja ratkaisun muotoileminen. Valmistelujakson aikana oppilas yrittää ymmärtää, mikä ongelma on ja miten sen voisi muotoilla mahdollisimman ymmärrettävästi ja täsmällisesti. Tutkivassa oppimisessa aivoriihi on osa valmistelujaksoa. Sen perusperiaatteena on nostaa esiin erilaisia ongelmaan liittyviä asioita perustelematta niiden merkitystä ratkaisun kannalta [12, s. 217]. Sen tarkoituksena on luoda pohja ongelman ymmärtämiselle ja ratkaisun keksimiselle. Tätä seuraa kypsymisjakso, jolloin alitajunnassa kehittyy erilaisia ideoita, joita analysoidaan ja vertaillaan. Kypsymisjaksoa seuraa niin sanottu inspiraatio, jolloin oppilas yhtäkkiä keksii ratkaisun ongelmaan. Lopuksi ratkaisu pitää tarkistaa ja kirjoittaa ymmärrettävästi ja täsmällisesti. Tutkivan oppimisen tarkoituksena on paitsi ratkaista ongelmia myös oppia soveltamaan ratkaisua tuleviin samankaltaisiin ongelmiin.

Ihmisillä on hyvin erilaisia tapoja ajatella luovasti. Silti luovaa ajattelua voidaan luonnehtia erilaisilla ominaisuuksilla. Luova ajattelu voi olla ongelmaherkkyyttä eli kykyä havaita ongelmia ja analysoida toisten ihmisten ajattelua. Se voi olla myös virtaavaa ajattelua, joka on esimerkiksi kykyä löytää ongelmaan useita eri ratkaisuja ja kykyä löytää tiedoille uusia sovelluskohteita. Ajattelun joustavuus on kykyä irrottautua tavanomaisesta ajattelusta ja löytää uusia kulmia ajatteluun. Luova ajattelu on myös omaperäistä ajattelua eli uusien vaihtoehtojen, joita muut eivät ole keksineet, tarjoamista sekä yleistämistä ja analogian käyttötaitoa, mikä on kykyä siirtää tietoa tilanteeseen, jossa siitä ei intuitiivisesti näyttäisi olevan mitään hyötyä. Luovan ajattelun perusta on keskittymiskyky. Etenkin edellä mainitut valmistelujakso ja kypsymisjakso vaativat ongelmanratkaisijalta keskittymiskykyä. [18, s.9-10]

Luova ajattelu kytkeytyy tutkivaan oppimiseen, joka on oppista omien kokemusten ja tutkimusten kautta. Tutkiva oppiminen kehittää oppilaan luovaa ajattelua ja ongelmanratkaisukykyä. Tutkiva oppiminen sisältää hyvin samankaltaisia vaiheita kuin luova prosessi. Siinä on kolme vaihetta. Ensimmäisessä vaiheessa oppilas tutkii ongelmaa ja pyrkii keksimää erilaisia ideoita ongelman ratkaisemiseksi [18, s.11-13]. Tässä vaiheessa oppilaat asettavat itse ongelmaan kysymyksiä. Oppilaiden itse asettamat kysymykset ovat etenkin käsitteellisen ymmärryksen kehittymisen kannalta tärkeitä [12, s.212]. Toisessa vaiheessa opettaja ja oppilaat keskustelevat oppilaiden ideoista ja ajatuksista. Tässä vaiheessa opettajan olisi hyvä tarkkailla oppilaiden keskustelua ja johdatella tarvittaessa oppilaita oikeaan suuntaan sekä vastata oppilaiden kysymyksiin. Viimeinen vaihe on käytäntö, joka on tutkitun asian soveltamista käytännössä. Tässä vaiheessa oppilaan on hyvä saada palautetta työskentelystä, sillä opitun asian arviointi on osa oppimista ja se auttaa oppilasta kehittämään ymmärrystään

opittua asiaa kohtaan.

Tutkivaa oppimista voidaan järjestää monin eri tavoin, mutta opettajan olisi hyvä huomioida, että oppilaat keksivät ratkaisuita hyvin eritahtiin. Tutkiva oppiminen on usein hitaampaa verrattuna perinteiseen opetuksen. Tavallisesti keksivää oppimista sovelletaan kouluissa erilaisin tavoin spontaanista keksimisestä ohjelmoituun keksimiseen. Spontaani keksiminen on hyvin sattumanvaraista oppilaiden omiin ajatuksiin pohjautuvaa tutkimista. Opettaja voi myös olla aloitteen tekijä, jolloin opettaja luo oppilaille tilanteen, jossa he voivat joko vapaasti tutkia ilmiöitä tai opettajan määräämän tutkimuksen kulun mukaan. Opettaja voi myös ennakolta suunnitella tutkimuksen, jolloin opettaja kontrolloi oppilaiden keksimistä ja oppilaat päätyvät hyvin todennäköisesti opettajan suunnittelemaan loppuratkaisuun. [18, s.11-13]

2.4 Kielentäminen

Kielentäminen on matemaattisen ajattelun esittämistä luonnollisen kielen, matemaattisen symbolikielen tai kuvakielen avulla [16, s.2-3]. Matematiikkaan on historian saatossa kehittynyt oma kielensä, jolla on omia erityispiirteitä. Matematiikan kielessä käytetään osittain muusta kielestä tuttua sanastoa kuten jatkuva tai rengas uudessa merkityksessä. Lisäksi matematiikan kieli sisältää huomattavasti omaa erikoissanastoa kuten integraali ja derivaatta ja siinä käytetään erityisellä tavalla normikielen puhetapoja [28, s.152-153]. Matematiikan esittäminen luonnollisen kielen avulla kehittää oppilaan matemaattista ymmärrystä ja parantaa oppimistuloksia. Lisäksi se auttaa opettajaa arvioinnissa ja parantaa oppilaiden asennetta matematiikan opiskelua kohtaan [16, s.2-3].

Matematiikan kuten muidenkin kielten oppiminen voi tapahtua passiivisesti tai aktiivisesti. Passiivista oppimista tapahtuu esimerkiksi, kun opettaja käyttää matematiikan kieltä täsmällisesti opettaessaan oppilaita. Aktiivista oppimista tapahtuu oppilaiden käyttäessä kieltä itse. Kuten vieraiden kielten opiskelussa matematiikan kielenkään opetuksessa opiskelija ei voi saavuttaa aktiivista kielitaitoa pelkän passiivisen oppimisen kautta. Silfverbergin, Portaankorva-Koiviston ja Yrjänäisen mukaan matematiikan didaktiikka on osittain myös kielididaktiikkaa. [28, s.153]

Alakoulussa matematiikan kielentäminen tarkoittaa matematiikassa ratkaisujen esittämistä symbolein, luonnollisen kielen avulla ja kuvakielellä. Perinteisesti oppilaat esittävät ratkaisun esimerkiksi paperilla matemaattisten symbolien kuten numeroiden ja aritmeettisten symbolien avulla. Kielentäminen tarkoittaa ratkaisun esittämistä matemaattisen symbolikielen lisäksi luonnollisen kielen avulla, mikä tarkoittaa ratkaisun esittämistä esimerkiksi kertomalla suullisesti tai kirjallisesti ratkaisu vaihe vaiheelta. Lisäksi ratkaisua voidaan havainnollistaa kuvakielen avulla eli piirtämällä tai muuten visuaalisesti esittämällä. Kielentämisen tavoitteena on kehittää oppilaan kykyä kertoa omista päättelyprosesseista ja siten syventää oppilaan ymmärrystä opituista asioista.

3 Nelikulmiot ja ohjelmointi opetussuunnitelmassa sekä aikaisemman tutkimukset

Geometria on ollut tärkeä osa matematiikkaa jo antiikin ajoista lähtien. Sen asema peruskoulun matematiikassa on edelleen tärkeä. Sen sijaan ohjelmointi on tullut osaksi perusopetuksen tavoitteita vasta viimeisimmässä opetussuunnitelmassa. Luvussa 3.1 käsitellään nelikulmioiden ja ohjelmoinnin asemaa peruskoulun opetussuunnitelmassa. Opetussuunnitelmassa ohjelmointi on kytketty pitkälti osaksi matematiikkaa. Matematiikan ja ohjelmoinnin yhdistämistä opetuksessa on kuitenkin tehty erittäin vähän tutkimusta. Luvussa 3.2 käsitellään matematiikan opetusta ohjelmoinnin avulla käsitteleviä aikaisempia tutkimuksia.

3.1 Nelikulmiot ja ohjelmointi opetussuunnitelmassa

Tähän tutkimukseen liittyvät teemat nousevat viimeisimmästä opetussuunnitelmasta. 2014 vuoden opetussuunnitelmaan sisällytettiin ohjelmoinnin opiskelu. Aikaisemmin ohjelmointia ei ole mainittu opetussuunnitelman tavoitteissa. Lisäksi opetussuunnitelman tavoitteisiin lisättiin muun muassa matematiikan kielentämistä sekä työelämätaitojen opettelua. Tässä luvussa tarkastellaan myös geometrian tavoitteita 3-6 luokille.

Opetussuunnitelma määrittelee yhdeksi oppimisen tavoitteeksi työelämätaitojen kehittämisen. Sen mukaan *Ajattelun taitoja harjoitellaan ongelmanratkaisu- ja päättelytehtävin sekä uteliaisuutta, mielikuvitusta, kekseliäisyyttä ja toiminnallisuutta hyödyntävin ja edistävin työskentelytavoin* [21, s.115]. Geometrian tutkiminen ohjelmoinnin avulla tarjoaa paljon mahdollisuuksia edellä mainittujen ajattelun taitojen harjoitteluun. Kaiken kaikkiaan tutkimuksessa kehitetty oppimateriaali on pyritty rakentamaan siten, että se tukee mahdollisimman hyvin opetussuunnitelmassa mainittuja ajattelun taitoja matematiikan oppimistavoitteiden osalta. Opetussuunnitelman keskeiset taidot on esitetty seuraavasti: *"Kehitetään oppilaiden taitoja löytää yhtäläisyyksiä, eroja ja säännönmukaisuuksia. Syvennetään taitoa vertailla, luokitella ja asettaa järjestykseen, etsiä vaihtoehtoja systemaattisesti, havaita syy- ja seuraussuhteita sekä yhteyksiä matematiikassa. Suunnitellaan ja toteutetaan ohjelmia graafisessa ohjelmointiympäristössä"* [21, s.235].

Opetussuunnitelman mukaan 3-6 luokilla on tavoitteena *"ohjata oppilasta havainnoimaan ja kuvailemaan kappaleiden ja kuvioiden geometrisia ominaisuuksia sekä tutustuttaa oppilas geometrisiin käsitteisiin"*[21, s.235]. Nelikulmioiden tarkemmalla tarkastelulla tarkoitetaan muun muassa nelikulmioiden luokittelua. Nelikulmiot voidaan jakaa erilaisiin ryhmiin niiden ominaisuuksien perusteella ja ryhmien välillä voidaan muodostaa hierarkioita.

Alakoulun 3-6 luokilla tavoitteena on *"innostaa oppilasta laatimaan toimintaohjeita tietokoneohjelmina graafisessa ohjelmointiympäristössä"* [21, s.235]. Alakoulussa tavoitteena on innostaa oppilaita ohjelmoinnin pariin ja opetella algoritmisen

ajattelun perusteita. Ohjelmoinnin ajattelu on sisällytetty osaksi matematiikan opetusta, joten voi olla luontevaa sitoa se osaksi muuta matematiikan opetusta. Alakoulussa ohjelmointia voidaan soveltaa esimerkiksi geometrian tarkasteluun ja tutkimiseen. Sen sijaan algoritmisen ajattelun kehittäminen ja ohjelmoinnin soveltaminen ongelmien ratkaisuun sisältyy vasta yläkoulun tavoitteisiin [21, s.375].

Opetussuunnitelman haasteena on se, että matematiikan oppisisältöjä on paljon ja lisäksi uutena tavoitteena on tullut ohjelmoinnin perusteiden opetus. Toistaiseksi on olemassa vain vähän suomenkielistä materiaalia ohjelmoinnin opetuksen tueksi. Etenkin materiaalia, joka sitoo ohjelmoinnin opetuksen muihin matematiikan oppisisältöihin, on olemassa erittäin vähän. Alakoulussa ohjelmoinnin opetus keskittyy loogisten toimintaohjeiden muodostamiseen ja mahdollisesti graafisten ohjelmointikielien opetteluun. Varsinaista lausekielistä tai muuta vastaavaa ohjelmointia ei alaluokilla opetussuunnitelman mukaan tarvitse opettaa.

3.2 Aikaisemmat tutkimukset

Matematiikan opiskelua ohjelmoinnin avulla on tutkittu erittäin vähän. Förster on tutkinut geometrian opiskelua Scratchin avulla kuudennella ja seitsemännellä luokalla Saksassa. Tutkimukseen osallistui neljä luokkaa, joissa oli yhteensä 98 oppilasta. Kolme luokkaa opiskeli kolmioiden geometriaa perinteisesti ja yksi luokka opiskeli samat asiat Scratchin avulla. Oppituntien jälkeen kaikille luokille pidettiin sama testi kolmioista. Testi tehtiin perinteisesti kynällä ja paperilla. Tutkimustulosten mukaan luokka, joka opiskeli geometriaa ohjelmoinnin avulla, suoriutui testistä huomattavasta paremmin verrattuna muihin luokkiin. Tutkimustulosten mukaan ohjelmointi syvensi selvästi oppilaiden geometrian ymmärrystä. [10, s.93-95]

Joki [13, s.96-97] on tutkinut nelikulmioiden hierarkian opetusta alakoulussa. Hänen mukaansa kuperien nelikulmioiden luokittelu on liian raskas alakoulun alemmille luokille. Sen sijaan hän ehdottaa hierarkian opetuksen siirtämistä peruskoulun kolmelle viimeiselle luokalle. Joki ehdottaa tutkimuksessaan opastamaan oppilasta sisäkkäisten luokkien systematiikkaan. Tällöin oppilaan käsitystä pyritään ohjaamaan siten, että hän hahmottaa nelikulmiot seuraavasti. Kaikilla nelikulmioilla on neljä sivua, mutta jotkin näistä nelikulmioista ovat erikoistapauksia. Nelikulmio, jonka vastakkaiset sivut ovat yhdensuuntaisia on suunnikas. Suunnikas, jonka kaikki kulmat ovat suoria, on erityinen suunnikas ja sitä kutsutaan suorakulmioksi. Suorakulmio, jonka kaikki sivut ovat yhtä pitkät, on erityinen suorakulmio nimeltään neliö.

Silfverbergin [27, s.170-174] tutkimuksen perusteella sisäkkäiset luokat vaikuttavat olevan oppilaille vaikeita ymmärtää. Tutkimuksen mukaan kaikilla yläkoulun luokka-asteilla oppilailla oli vaikeuksia hahmottaa erilaisten nelikulmioiden välinen hierarkia. Suurin osa ymmärsi eri hierarkialuokkien lukeutuvan nelikulmioiksi, mutta suurin osa oppilaista ei ymmärtänyt muiden luokkien välisiä suhteita. Lähes kaikki oppilaat lukivat neliön ja suunnikkaan sekä suorakulmion ja suunnikkaan toisensa poissulkeviksi ominaisuuksiksi. Kaiken kaikkiaan lähes kaikilla oppilailla oli vaikeuksia hahmottaa nelikulmioiden käsitteiden suhteita.

4 Ohjelmointi

Ohjelmoinnilla tarkoitetaan prosessia, jossa kehitetään ja testataan joukkoa ohjelmoituja käskyjä [4]. Ohjelmoituja käskyjä toteutetaan erilaisilla ohjelmointikielillä. Ohjelmointikielet voidaan jakaa kielen paradigman mukaisiin ryhmiin. Paradigma on ohjelmointikielen keskeinen toteutusperiaate, joka määrittelee, millaisista komponenteista ohjelmisto rakentuu ja miten kontrolli siinä etenee. Paradigmoja ovat muun muassa olio-ohjelmointi ja funktionaalinen ohjelmointi. Ohjelmointikielet voivat toteuttaa yhtä tai useampaa paradigmaa. Useita paradigmoja yhdistelevää kieltä kutsutaan yleisesti moniparadigmaiseksi kieleksi.

Olio-ohjelmoinnissa olioita käytetään kuvaamaan abstrakteja tai reaalimaailman ilmiöitä. Olio-ohjelmointiin perustuvilla kielillä sovellusten ohjelmointi perustuu eri olioiden yhteistoimintaan. Olioperustaisia kieliä ovat esimerkiksi yleisesti käytössä olevat kielet Java ja C#. Funktionaalisissa kielissä kontrolli etenee funktioiden kautta. [19] Funktionaalisia kieliä ovat esimerkiksi Scala ja Scheme. Tällä hetkellä olio-ohjelmointi on käytetyin paradigma etenkin mobiililaitteiden sovellusten ohjelmoinnissa. Algoritmisen ajattelun ja ohjelmoinnin opetuksen tueksi kehitetyt graafiset ohjelmointikielet rakentuvat myös pitkälti olioperustaisen ohjelmoinnin päälle. Olio-ohjelmointi voi olla aloittelevalle ohjelmoijalle helpompi ymmärtää, koska siinä oliot voidaan kuvata konkreettisina reaalimaailman asioina. Esimerkiksi lapsille suunnatussa graafisessa ohjelmointiympäristössä Scratchissä oletusolio kuvataan kissana. Kissalle annetut käskyt on intuitiivisesti helppoa ymmärtää ja sitä kautta paradigman hahmottaminen alkaa huomaamatta kehittyä. Tässä tutkimuksessa on rajattu ohjelmointi käsittämään olio-ohjelmointia ja muut paradigmat sivuutetaan.

4.1 Algoritminen ajattelu

Algoritmiselle ajattelulle ei ole olemassa yksiselitteistä määritelmää. Ajan saatossa useat tutkijat ovat määritelleet algoritmisen ajattelun hieman eri tavoin. Pääasiassa määritelmiä yhdistää algoritmisen ajattelun kuvaaminen eräänlaisena ongelmanratkaisustrategiana. Osassa määritelmistä tietokone on keskeisessä asemassa algoritmisessa ajattelussa, kun taas osa määritelmistä korostaa ajatteluprosessia, jolla ongelma muotoillaan sopivaan muotoon. Algoritmista ajattelua on myös kritisoitu muun muassa määritelmän epäselvyyden ja laajuuden takia.

Simon Papert käytti ensimmäistä kertaa termiä algoritminen ajattelu (computional thinking) matematiikan opiskelua käsittelevässä artikkelissaan vuonna 1996. Papert kuvaa algoritmisen ajattelun olevan tietokoneen käyttöä ongelman ratkaisussa muodostamaan ideoita, joiden avulla ihmiset voivat kehittää parempia analyysejä, selittää ongelmia ja ratkaisuja sekä muodostaa niiden välille yhteyksiä [20, s. 2]. Wingin mukaan algoritminen ajattelu on ongelmanratkaisustrategia, jossa ongelma pyritään muotoilemaan siten, että tietokone tai muut vastaavat työkalut voivat ratkaista ongelman. Ongelmanratkaisija järjestää lähtötietoja loogisesti ja pyrkii esittämään ne erilaisten mallien simulaatioiden avulla. Ongelmanratkaisuprosessi pyritään tunnis-

tamaan ja automatisoimaan siten, että vastaavat ongelmat voidaan ratkaista samalla algoritmilla. Ratkaisua voidaan pyrkiä myös yleistämään moniin erilaisiin ongelmiin. Algoritminen ajattelu auttaa ratkaisemaan ongelmia ja luomaan systeemeitä, joita ihminen ei kykenisi luomaan ilman koneita. [24, s.131] Aho taas määrittelee algoritmisen ajattelun ajatteluprosessiksi, jossa ongelma pyritään esittämään laskennallisina askelina ja algoritmeina. Tärkeä osa prosessia on löytää laskennallinen malli, jonka avulla ratkaisu voidaan muotoilla ja ratkaista. [1, s. 2-3]

Papertin ja Wingin määritelmät algoritmisesta ajattelusta eroavat toisistaan perustavanlaatuisesti. Papert määrittelee algoritmisen ajattelun ideoiden muodostamisen ja selittämisen kautta, kun taas Wing määrittelee sen ongelman muotoilun kautta. Wingin määritelmässä tietokone ei ole välttämättä osa algoritmista ajattelua ja Papertin määritelmässä tietokone on keskeisessä asemassa. Erot määritelmissä aiheuttavat ongelmia etenkin tutkimuksen kannalta. ISTE (International Society for Technology in Education) on pyrkinyt ratkaisemaan ongelman kehittämällä toimintamääritelmän, joka kuvaa algoritmista ajattelua ongelmanratkaisuprosessiksi, jolla on seuraavia omaisuuksia: Ongelmien muotoilu siten, että tietokone tai muu työkalu voi ongelman ratkaista, looginen järjestäminen ja datan analysointi, datan esittäminen abstraktioiden kuten mallien ja simulaatioiden avulla, ratkaisujen automatisointi, mahdollisten ratkaisujen analysointi, tunnistaminen ja toteutus mahdollisimman tehokkaan ratkaisun löytämiseksi sekä ongelmanratkaisuprosessin yleistäminen ja soveltaminen vastaaviin ongelmiin [20, s. 3]. Tämä määritelmä on kehitetty erityisesti algoritmisen ajattelun opetuksen tueksi.

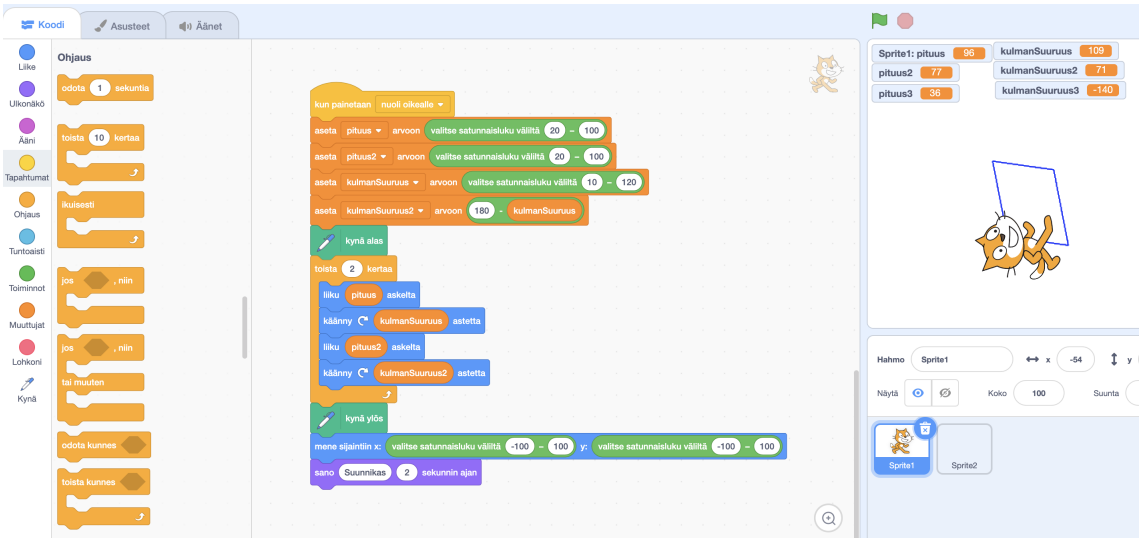
Algoritmisesta ajattelusta on esitetty myös kritiikkiä. Yleisimmät aiheet kritiikkiin ovat termin liian laaja määritelmä ja variaatio eri määritelmien välillä. Peter Denning on kritisoinut algoritmista ajattelua siitä, että se on uusi kapeampi termi tietojenkäsittelytieteille (computer science). Termiä on kritisoitu myös siitä, että se korostaa liiaksi algoritmista ajattelua algoritmien tekemisen ja opettelun sijaan. Kritiikki kohdistuu erityisesti siihen, että algoritmista lähestymistapaa pyritään korostamaan liiaksi ja tuomaan osaksi muita tieteenaloja, joissa on perinteisesti käytetty muita lähestymistapoja. [20, s. 3]

Alakoulun matematiikassa opetellaan algoritmisen ajattelun perusteita. Pääasiassa oppilaat opettelevat erilaisten suhteellisen lyhyiden toimintaohjeiden laatimista ohjelmistoympäristössä tai luokkatilassa ilman tietokoneita. Toimintaohjeiden laatimisella pyritään kehittämään oppilaiden taitoja järjestää ongelman lähtötietoja loogiseen järjestykseen siten että ohjelmointikieli osaa tulkita ratkaisun halutulla tavalla. Alakoulussa voidaan luoda myös yksinkertaisia algoritmeja ja ohjelmia graafisilla ohjelmointikielillä, mutta opetussuunnitelman mukaan niiden laatiminen on vasta yläkoulun matematiikassa tavoitteena. Lausekielisiin ohjelmointikieliin tutustuminen on tavoitteena vasta yläkoulun puolella.

4.2 Graafiset ohjelmointikielet

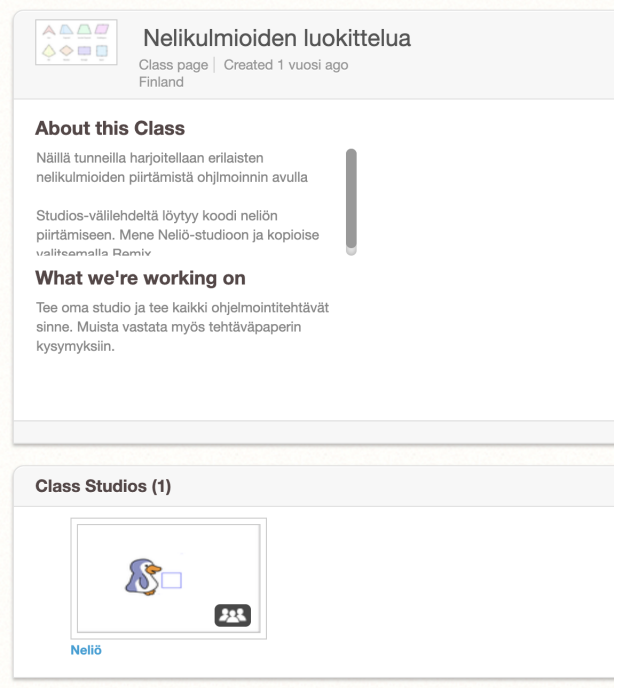
Graafisia ohjelmointikieliä on kehitetty viime aikoina etenkin lapsille ja nuorille ensimmäiseksi ohjelmointikieleksi. Graafiset ohjelmointikielet tarjoavat nuorille oppilaille mahdollisuuden ohjelmoida omia ohjelmia opettelematta perinteisten ohjelmointikielien monimutkaisia syntakseja. Niiden avulla oppilaat voivat kehittää algoritmisen ajattelun taitoja ilman perinteisten ohjelmointikielten esteitä [6, s.1]. Graafiset ohjelmointikielet eroavat muista ohjelmointikielistä siten, että niissä ei ole lausekielistä syntaksia, vaan se on korvattu lohkoilla. Yksi lohko kuvaa yhtä ohjelmointikielen toimintoa. Lapsille ja nuorille suunnatut ohjelmointikielet ovat käytettävissä tietyssä ohjelmointiympäristössä. Ohjelmointiympäristö tarkoittaa esimerkiksi verkkosivustoa tai sovellusta, joka tarjoaa mahdollisuuden toteuttaa ohjelman rajatussa ympäristössä.

Aloittelijoille suunnatuissa graafisista ohjelmointikielistä löytyvät lähtökohtaisesti kaikki olioperustaisten ohjelmointikielien perusrakenteet ja ominaisuudet. Näin ollen kielet on suunniteltu tukemaan ohjelmoinnin rakenteiden oppimista ja ohjelmoinnillisen ajattelun kehittymistä. Graafisia ohjelmointikieliä on luotu sekä oppimisen tueksi, että innostamaan lapsia ja nuoria ohjelmoinnin pariin. Graafisten ohjelmointikielien etuna on se, että niiden avulla voidaan luoda näyttäviä sovelluksia suhteellisen lyhyessä ajassa. Weindropin ja Wilenskyn tekemän tutkimuksen mukaan graafisella kielellä ohjelmoivat opiskelijat olivat innokkaampia opiskelemaan lisää ohjelmointia kuin oppilaat, jotka opiskelivat lausekielistä ohjelmointia. Tutkimuksessa graafisella kielellä ohjelmoivat opiskelijat menestyivät kokeessa hieman paremmin verrattuna lausekielellä ohjelmoiviin opiskelijoihin, vaikka kokeessa syntaksi oli tutumpi lausekielellä ohjelmoiville opiskelijoille. [35, s.17-18]



Kuva 4.1: Scratchillä tehty onjelma, jossa on käytetty olio-ohjelmoinnin peruselementtejä.

Tutkimuksessa on valittu graafiseksi ohjelmointikieleksi Scratch. Scratch on opetustarkoitukseen kehitetty graafinen ohjelmointiympäristö. MIT Media Lab on kehittänyt sen erityisesti lapsille ja nuorille ensimmäiseksi ohjelmointiympäristöksi [11, s.10]. Scratch on kehitetty olioperustaisen Java-kielen pohjalta ja sen valikoimasta löytyvät kaikki yleisimmät Javan ominaisuudet. Sen valikoimassa on myös runsaasti laajennoksia kuten äänituki, Google kääntäjä ja tuki robottien ohjaukseen. Scratchissä oliot kuvaavat lähtökohtaisesti reaalimaailman asioita, mikä tekee olioperustaisuuden helpommaksi ymmärtää. Toisaalta Scratch tukee myös jonkin verran abstrakteja olioita kuten listoja. [26]. Suurelle osalle ohjelmointia opettelevista oppilaista Scratch tai sen pienille lapsille suunnattu kieli ScratchJr on ensimmäinen ohjelmointikieli. Etenkin niiden oppiaineiden tunneilla, joiden tavoitteet eivät liity suoraan tietotekniikkaan, Scratch on hyödyllinen työkalu algoritmiseen ajattelun kehittämiseen [6, s.2]. Scratch onkin kehitetty ensisijaisesti innostamaan lapsia ja nuoria ohjelmoinnin pariin ja kehittämään oppilaiden algoritmista ajattelua, jotta lausekielisen ohjelmoinnin aloittaminen olisi helpompaa. Lausekielisessä ohjelmoinnissa on haasteena vaikean syntaksin lisäksi se, että lausekielisellä ohjelmoinnilla on erittäin haastavaa luoda visuaalisesti näyttäviä ohjelmia. Scratch mahdollistaa erittäin hyvin näyttävien visuaalisten ohjelmien kuten pelien ja animaatioiden luomisen, mikä motivoi oppilaita. Hyvä motivaatio auttaa erittäin paljon siirtymässä graafisesta ohjelmoinnista lausekieliseen ohjelmointiin.



Kuva 4.2: Oppilaan luokkanäkymä.

Scratch sopii hyvin ohjelmoinnin opetukseen alakoulussa, koska ohjelma on helppokäyttöinen ja selkeä. Lisäksi ohjelmoinnin logiikka vastaa lausekielistä ohjelmointikieltä, mutta oppilaiden ei tarvitse keskittyä syntaksiin ja siitä johtuviin virheisiin. Scratch on ilmainen ja sitä voidaan käyttää tietokoneen verkkoselaimella, joten ohjelmoinnin aloittaminen on helppoa eikä sen käyttö ole riippuvainen laitteesta tai käyttöjärjestelmästä. Sitä on mahdollista käyttää myös mobiililaitteilla. Scratch mahdollistaa omien ohjelmien luomisen yksinkertaisten toimintaohjeiden avulla. Sillä voi luoda yksinkertaisia toimintaohjeita sisältäviä ohjelmia sekä monimutkaisempia pelejä ja animaatioita rajatussa ympäristössä.

Scratch on suunniteltu opetuskäyttöön, joten siinä on paljon opetusta tukevia elementtejä. Opettaja voi luoda luokkahuoneen sekä hallita ja luoda oppilaiden käyttäjätunnuksia. Lisäksi Scratch tarjoaa laajan materiaalipankin, joka on sekä oppilaiden että opettajien käytettävissä. Scrach on tällä hetkellä ainoa ohjelmointikieli, jonka avulla ohjelmointi onnistuu suomen kielellä. Etenkin alakoululaisille tämä on tärkeää, jotta puutteet kielitaidossa eivät heikentäisi oppimistuloksia ja -motivaatiota.

Scratchissä opettajalla on mahdollisuus luoda opettajan käyttäjätili. Tilillä opettaja voi luoda luokkahuoneita ja esimerkiksi luoda oppilaille tunnuksia ohjelmaan, jolloin oppilaiden ei tarvitse antaa henkilötietojaan ohjelman käyttöön. Opettaja voi luoda luokkahuoneita, joiden sisällä oppilaat voivat muun muassa saada tehtävänantoja ja jakaa omia tuotoksiaan muun luokan tarkasteltavaksi. Opettaja voi myös tarkastella oppilaiden töitä ja tarvittaessa kommentoida niitä. Opettaja voi myös sulkea luokkahuoneen opetuksen päätyttyä, jolloin oppilaiden tunnukset ja niihin liittyvät tiedot poistuvat järjestelmästä. Tämä toiminto sopii erinomaisesti myös tutkimuksen suorittamiseen, koska tutkijan on helppoa tarkastella aineistoa ja tutkimuksen päätyttyä poistaa se.

5 Nelikulmiot geometriassa

Tässä luvussa määritellään nelikulmio ja erilaisten nelikulmioiden hierarkia matemaattisesti. Määritelmiä varten käsitellään aluksi euklidisia avaruuksia, joissa nelikulmiot voidaan määritellä. Peruskoulun matematiikassa geometria perustuu euklidisiin avaruuksiin. Alakoulussa opiskellaan vain euklidisen tason geometriaa, joten tässä tutkimuksessa tutkitaan nelikulmioita erityisesti euklidisen tason ilmiöinä. Lohkoperustaiset opetuskäyttöön suunnitellut ohjelmointiympäristöt perustuvat sekä käyttöliittymältään, että ohjelmointilogiikaltaan euklidisen tason ominaisuuksiin. Siksi tässä tutkimuksessa geometristen avaruuksien tarkastelua on rajattu käsittämään alakoulun matematiikan ja graafisten ohjelmointiympäristöjen määrittämiseen tarvittavaa matematiikkaa.

Aluksi käsitellään lyhyesti euklidisten avaruuksien historiaa luvussa 5.1. Jotta euklidinen avaruus voidaan määritellä, tarvitaan vektoriavaruuden määritelmä (luku 5.2). Vektoriavaruuden avulla määritellään euklidinen avaruus sekä muutamia tarpeellisia euklidisen avaruuden ominaisuuksia. Tämän jälkeen käydään läpi Eukleideen aksioomat ja määritellään monikulmio. Lopuksi määritellään vielä joukko erilaisia nelikulmioita sekä niiden välinen hierarkia.

5.1 Euklidisen avaruuden historiaa

Geometria on yksi vanhimmista matematiikan aloista. Geometria käsittelee erilaisia kuvioita ja kappaleita sekä niiden ominaisuuksia. Geometrian uskotaan kehittyneen erityisesti maanmittauksen seurauksena. Maanmittausta kehitettiin aluksi Egyptistä. Egyptistä maanmittaustaito siirtyi antiikin Kreikkaan, missä sitä alettiin kehittää. Maanmittaus oli tärkeää sekä Egyptissä että Kreikassa veronmaksun kannalta.

Euklidinen avaruus on saanut nimensä kreikkalaisen matemaatikon Eukleides Aleksandrialaisen mukaan. Eukleides eli antiikin Kreikassa n. 300 eaa. Eukleides tunnetaan parhaiten hänen julkaisemastaan kirjasta Elementa, joka on suomennettu nimellä Alkeet. Kirjassa hän käsittelee geometrian aksioomia sekä niistä johdettuja tuloksia. Lisäksi Eukleides käsittelee kirjassaan aritmetiikkaa ja lukuteoriaa sekä deduktiiviseen päättelyyn perustuvaa todistamista. Eukleideen aksioomia käytetään edelleen ja aksioomat toteuttavia avaruuksia kutsutaan euklidisiksi avaruuksiksi. Eukleidein määritelmän mukaan seuraavien ehtojen tulee täyttyä euklidisessa avaruudessa.

1. Voidaan vetää suora viiva mistä pisteestä tahansa mihin pisteeseen tahansa. Tässä Eukleides tarkoittaa kahden pisteen kautta kulkevaa suoraa, joka voidaan aina määrittää yksikäsitteisesti.

2. Janaa voidaan jatkaa kumpaankin suuntaan yli päätepisteistänsä.

3. Pisteen ympäri voidaan piirtää minkä tahansa toisen pisteen kautta kulkeva ympyrä.

4. Kaikki suorat kulmat ovat yhtä suuria.

5. Olkoot ℓ suora ja P piste, joka ei ole suoralla ℓ. Tällöin on olemassa tasan yksi suora, joka kulkee pisteen ℓ kautta leikkaamatta suoraa ℓ. Tämä suora on yhdensuuntainen suoran ℓ kanssa. (Paralleeliaksiooma) (vrt. [8, s. 14])

Eukleideen aksioomat ovat olleet geometrian perustana 1800-luvulle asti. 1800-luvulla useamman matemaatikon onnistui osoittaa, että on olemassa avaruus, jolla Eukleideen paralleeliaksiooma ei päde. Näitä avaruuksia alettiin kutsua epäeuklidisiksi avaruuksiksi. Edelleenkin suomalaisessa peruskoulussa ja lukiossa käsitellään vain euklidisia avaruuksia. Tässä tutkimuksessa keskitytäänkin euklidisiin avaruuksiin ja niissä määriteltyihin kuvioihin sekä tarkastellaan erityisesti euklidista tasogeometriaa. Lapsille suunnattujen graafisten ohjelmointikielien logiikka ja käyttöliittymä perustuvat euklidiseen tasoon. [29], [14]

5.2 Vektoriavaruus

Tässä luvussa esitetään vektoriavaruuden määritelmä, jota tarvitaan euklidisen avaruuden määrittelyyn. (vrt. [2, s. 2])

V on vektoriavaruus, jos V on epätyhjä joukko, jossa on määritelty alkioiden summa $+ : V \times V \to V$ ja skalaarikertolasku $\mathbb{R} \times V \to V$. Lisäksi seuraavat ehdot toteutuvat:

1. $X + Y = Y + X$ kaikilla $X, Y \in V$. (vaihdannaisuus)

2. $(X + Y) + Z = X + (Y + Z)$ kaikilla $X, Y, Z \in V$. (liitännäisyys)

3. On olemassa $\mathbf{0} \in V$, jolle pätee $X + \mathbf{0} = X$ kaikilla $X \in V$. Tätä kutsutaan nolla-alkioksi.

4. kaikilla $X \in V$ on olemassa sellainen $-X \in V$, jolle pätee $X + (-X) = 0$. Tätä kutsutaan vasta-alkioksi.

5. $a(X + Y) = aX + aY$ kaikilla $a \in \mathbb{R}$ ja $X, Y \in V$

6. $(a + b)X = aX + bX$ kaikilla $a, b \in \mathbb{R}$ ja $X \in V$

7. $a(bX) = (ab)X$ kaikilla $a, b \in \mathbb{R}$ ja $X \in V$

8. $1X = X$ kaikilla $X \in V$.

5.3 Euklidiset avaruudet

Määritellään seuraavaksi euklidinen avaruus. Euklidinen avaruus $\mathbb{R}^n, n \in \mathbb{N}$ on vektoriavaruus

$$\mathbb{R}^n = \{(x_1, x_2, ..., x_n) | x_i \in \mathbb{R} \text{ kaikilla } i \leq n\}$$

Olkoon X ja $Y \in \mathbb{R}^n$. Määritellään vektorien summa ja skalaarikertolasku euklidisessa avaruudessa seuraavasti. Summa on

$$X + Y = (x_1 + y_1, x_2 + y_2, ..., x_n + y_n)$$

ja skalaarikertolasku

$$rX = r(x_1, x_2, ..., x_n) = (rx_1, rx_2, ..., rx_n), \text{ missä } r \in \mathbb{R}.$$

Määritellään nolla-alkioksi $\mathbf{0} = (0, 0, .., 0) \in \mathbb{R}^n$ ja vasta-alkioksi $-X = (-x_1, -x_2, ..., -x_n)$.

Lause 5.1. *Euklidinen avaruus on vektoriavaruus.*

Todistus. Osoitetaan euklidisen avaruuden olevan vektoriavaruus osoittamalla, että kaikki vektoriavaruuden määritelmän (5.2) kohdat täyttyvät. Olkoon $X = (x_1, x_2, ..., x_n)$, $Y = (y_1, y_2, ..., y_n)$ ja $Z = (z_1, z_2, ..., z_n)$ avaruuden $\mathbb{R}^n$ vektoreita. Lisäksi olkoot $a, b \in \mathbb{R}$

1. Osoitetaan aluksi, että vektorien yhteenlaskun vaihdannaisuus pätee avaruudessa $\mathbb{R}^n$
$$\begin{aligned} X + Y &= (x_1 + y_1, x_2 + y_2, ..., x_n + y_n) \\ &= (y_1 + x_1, y_2 + x_2, ..., y_n + x_n) \\ &= Y + X \end{aligned}$$

2. Osoitetaan seuraavaksi vektorien yhteenlaskun liitännäisyys
$$\begin{aligned} (X + Y) + Z &= (x_1 + y_1, x_2 + y_2, ..., x_n + y_n) + (z_1, z_2, ..., z_n) \\ &= (x_1 + y_1 + z_1, x_2 + y_2 + z_2, ..., x_n + y_n + z_n) \\ &= (x_1, x_2, ..., x_n) + (y_1 + z_1, y_2 + z_2, ..., y_n + z_n) \\ &= X + (Y + Z) \end{aligned}$$

3. Osoitetaan, että $\mathbf{0} \in \mathbb{R}^n$ on nollavektori. Nyt
$$\begin{aligned} X + \mathbf{0} &= (x_1 + 0, x_2 + 0, ..., x_n + 0) \\ &= (x_1, x_2, ..., x_n) = X \end{aligned}$$
Siis 0 on avaruuden nolla-alkio.

4. Olkoot $-X = (-x_1, -x_2, ..., -x_n)$. Nyt
$$\begin{aligned} X + (-X) &= (x_1 + (-x_1), x_2 + (-x_2), ..., x_n + (-x_n)) \\ &= (x_1 - x_1, x_2 - x_2, ..., x_n - x_n) = 0 \end{aligned}$$
Siis $-X$ on vektorin X vasta-alkio.

5. Osoitetaan seuraavaksi, että $a(X + Y) = aX + aY$

$$\begin{aligned} a(X + Y) &= a(x_1 + y_1, x_2 + y_2, ..., x_n + y_n) \\ &= (a(x_1 + y_1), a(x_2 + y_2), ..., a(x_n + y_n)) \\ &= (ax_1 + ay_1, ax_2 + ay_2, ..., ax_n + ay_n) \\ &= aX + aY \end{aligned}$$

6. Lisäksi vektoriavaruudelle on oltava $(a + b)X = aX + bX$.

$$\begin{aligned} (a + b)X &= ((a + b)x_1, (a + b)x_2, ..., (a + b)x_n) \\ &= (ax_1 + bx_1, ax_2 + bx_2, ..., ax_n + bx_n) \\ &= aX + bX \end{aligned}$$

7. Osoitetaan, että $a(bX) = (ab)X$

$$\begin{aligned} a(bX) &= a(bx_1, bx_2, ..., bx_n) \\ &= (abx_1, abx_2, ..., abx_n) \\ &= (ab)X \end{aligned}$$

8. Osoitetaan vielä, että $1X = X$.

$$\begin{aligned} 1X &= (1x_1, 1x_2, ..., 1x_n) \\ &= (x_1, x_2, ..., x_n) = X \end{aligned}$$

Koska kaikki kohdat 1.-8. pätevät, euklidinen avaruus $\mathbb{R}^n$ on todellakin vektoriavaruus. □

Määritelmä 5.1 (vrt. [9, s. 38])**.** Määritellään seuraavaksi vektorin $X = (x_1, x_2, ..., x_n)$ normi vektoriavaruudessa $\mathbb{R}^n$.

$$|X| = \sqrt{x_1^2 + x_2^2 + ... + x_n^2}$$

Määritelmä 5.2 (vrt. [9, s. 2])**.** Määritellään vektoreille $X = (x_1, x_2, ...x_n)$ ja $Y = (y_1, y_2, ..., y_n)$ metriikka eli etäisyysfunktio $d(X, Y)$ vektoriavaruudessa $\mathbb{R}^n$ seuraavasti

$$d(X, Y) = |X - Y| = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2}.$$

Määritelmä 5.3 (vrt. [9, s. 39])**.** Määritellään pistetulo $X \cdot Y$ vektoriavaruudessa $\mathbb{R}^n$ seuraavasti. Olk $X = (x_1, x_2, ...x_n), Y = (y_1, y_2, ..., y_n)$, missä $X, Y \in \mathbb{R}^n$

$$X \cdot Y = x_1y_1 + x_2y_2 + ... + x_ny_n$$

Peruskoulun matematiikassa käsitellään vain euklidisia avaruuksia $\mathbb{R}^2$ ja $\mathbb{R}^3$. Tässä tutkimuksessa käsitellään kaksiulotteisia avaruuksia $\mathbb{R}^2$, joten rajataan jatkossa avaruuksien käsittely avaruuteen $\mathbb{R}^2$.Tästä avaruudesta käytetään nimitystä Eukleideen taso.

5.3.1 Kulma ja suora euklidisessa avaruudessa

Tässä kappaleessa määritellään joitakin peruskäsitteitä euklidisessa tasossa.

Olkoon $A = (a_1, a_2)$ ja $B = (b_1, b_2)$ euklidisen tason $\mathbb{R}^2$ pisteitä. Pisteiden A ja B kautta kulkevaa suoraa viivaa kutsutaan suoraksi, jos se jatkuu rajattomasti molempiin suuntiin. Tällöin suoraa kutsutaan pisteiden A ja B kautta kulkevaksi suoraksi. Mikäli A ja B ovat viivan päätepisteitä, viivaa kutsutaan janaksi AB. (vrt. [32, s. 2])

Eukleideen tasossa etäisyys ja normi määritellään seuraavasti. Olkoot $X = (x_1, x_2)$ ja $Y = (y_1, y_2)$ euklidisen tason vektoreita. Nyt Vektorin X normi on

$$|X| = \sqrt{x_1^2 + x_2^2} \text{ ja}$$

vektoreiden X ja Y välinen etäisyys on

$$|X - Y| = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}.$$

Määritelmä 5.4 (vrt. [9, s. 41])**.** Määritellään seuraavaksi kahden vektorin välinen kulma $\theta \in [\,0, \pi\,]$ euklidisessa avaruudessa.

$$\theta = \arccos \frac{X \cdot Y}{|X||Y|}$$

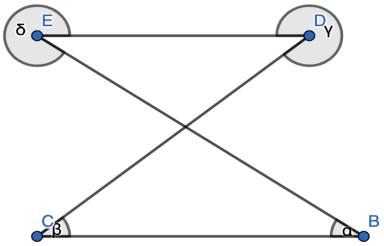
Vektoreiden X_1 ja Y_1 välinen kulma on yhtä suuri kuin vektoreiden X_2 ja Y_2 välinen kulma, jos lausekket $\dfrac{X_1 \cdot Y_1}{|X_1||Y_1|}$ ja $\dfrac{X_2 \cdot Y_2}{|X_2||Y_2|}$ saavat saman arvon. Kulmaa kutsutaan suoraksi kulmaksi, jos $\theta = \frac{\pi}{2}$.

5.3.2 Monikulmiot euklidisessa avaruudessa

Monikulmio euklidisessa avaruudessa määritellään pisteiden $(V_1, V_2, ..., V_p)$, missä $p \in \mathbb{N}$ ja $p > 2$, ja niitä yhdistävien janojen $(V_1V_2, V_2V_3, ...V_{p-1}V_p, V_pV_1)$ joukoksi. Pisteitä kutsutaan kärkipisteiksi ja janoja kuvion sivuiksi. Vierekkäisten janojen sisään jääviä kulmia kutsutaan monikulmion kulmiksi. Monikulmiossa on aina yhtä monta sivua ja kärkipistettä. Kuvio nimetään kärkipisteiden lukumäärän mukaan. Esimerkiksi, jos kuviossa on neljä kärkipistettä, kuvioa kutsutaan nelikulmioksi ja jos siinä on kuusi kärkipistettä, sitä kutsutaan kuusikulmioksi. [32, s.22], [9, s.69-70]

5.4 Nelikulmioiden määritelmiä ja hierarkioita

Nelikulmiot (quadrilateral) voidaan jakaa kahteen pääluokkaan, jotka ovat kompleksinen (complex) ja yksinkertainen (simple) nelikulmio. Euklidisessa tasossa kompleksinen nelikulmio leikkaa itsensä yhdessä pisteessä ja nelikulmion kahden vastakkaisen kulman suuruudet ovat yli $\frac{\pi}{2}$ ja toiset vastakkaiset kulmat ovat alle $\frac{\pi}{2}$. Yksinkertaisissa nelikulmioissa enintään yksi kulma on suuruudeltaan yli $\frac{\pi}{2}$. Peruskoulun matematiikassa käsitellään vain yksinkertaisia nelikulmioita, joten tässä tutkimuksessa ei tarkastella enempää kompleksisia nelikulmioita.



Kuva 5.1: Kompleksinen nelikulmio

Yksinkertaiset nelikulmiot jakautuvat edelleen kuperiin (convex) ja koveriin (concave) nelikulmioihin. Koverassa nelikulmiossa on yksi kulma, joka on suuruudeltaan yli $\frac{\pi}{2}$. Kuperissa nelikulmioissa kaikki neljä kulmaa ovat suuruudeltaan alle $\frac{\pi}{2}$. Peruskoulun matematiikassa ei käsitellä kuperan ja koveran nelikulmion eroa. Tyypillisesti peruskoulussa kupera ja kovera nelikulmio esitellään yleisenä nelikulmiona.

Kuperat nelikulmiot jakautuvat edelleen useisiin erilaisiin hierarkioihin. Hierarkialuokkien määritelmät ja lukumäärät vaihtelevat hieman lähteestä riippuen. Tässä luvussa käsitellään tarkemmin luokkia, jotka ovat käytössä suomalaisessa peruskoulussa. Näiden lisäksi on olemassa myös useita muita hierarkioita kuten leijat (kite) ja tasakylkiset trapetsit (isosceles trapezium). Leijassa yhdet vastakkaiset kulmat ovat yhtä suuret. Lisäksi vierekkäiset sivut ovat pareittain yhtä pitkät. Tasakylkisessä trapetsissa pätee puolisuunnikkaan määritelmä ja lisäksi sen vastakkaiset sivut ovat pareittain yhtä pitkät. Kuvassa 5.2 on esitetty nelikulmioiden hierarkia kokonaisuudessaan.

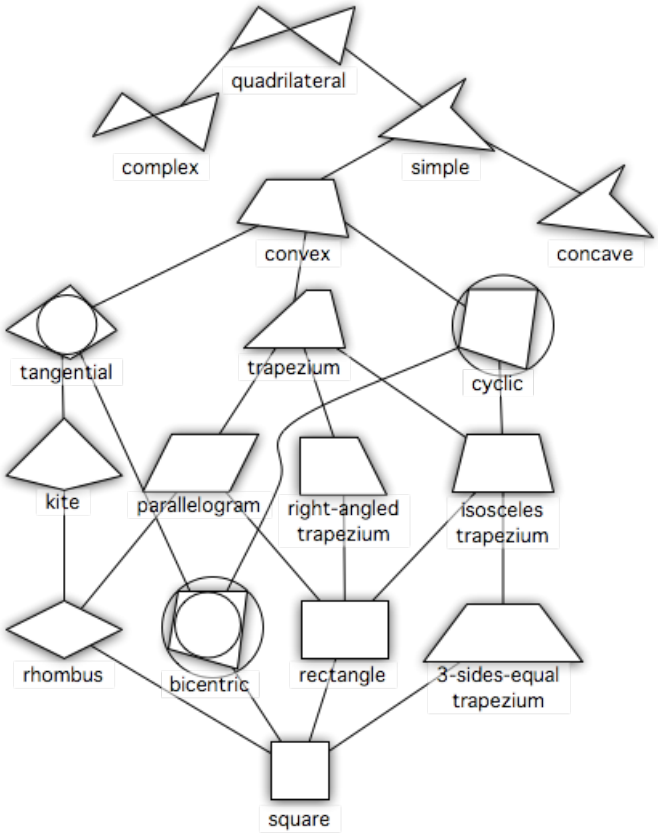
Suunnikas (parallelogram) on määritelmän mukaan nelikulmio, jossa vastakkaiset sivut ovat keskenään yhdensuuntaiset. Nelikulmio on suunnikas, jos ja vain jos vastakkaiset sivut ovat pareittain yhtä pitkät, vastakkaiset kulmat ovat pareittain yhtä suuret, nelikulmion lävistäjät puolittavat toisensa ja sillä on yksi symmetriakuvaus k_π. [31, s.23-26]

Puolisuunnikkaalla (trapezium) on olemassa kaksi määritelmää. Ensimmäisen määritelmän mukaan nelikulmio on puolisuunnikas, jos täsmälleen yhdet nelikulmion sivut ovat pareittain yhdensuuntaiset. Toisen määritelmän mukaan nelikulmio on puolisuunnikas, jos vähintään yhdet nelikulmion sivut ovat pareittain yhdensuuntaiset. Ensimmäinen määritelmä on käytössä Suomen kouluissa ja myös esimerkiksi Yhdysvalloissa. Toinen määritelmä on yleinen etenkin Keski-Euroopassa. Kuvassa 5.2 hierarkia on esitetty Keski-Eurooppalaisen määritelmän pohjalta. Puolisuunnikkaan määritelmä vaikuttaa olennaisesti nelikulmioiden hierarkiaan, sillä ensimmäisen määritelmän mukaan suunnikas ei ole puolisuunnikas, mutta toisen määritelmän mukaan suunnikas lukeutuu myös puolisuunnikkaisiin. Kuvasta 5.2 voidaan havaita, että suunnikas on luettu puolisuunnikkaan erikoistapaukseksi, kun se suomessa yleisesti käytössä olevan määritelmän mukaan luettaisiin kuperan nelikulmion erikoistapaukseksi, mutta ei puolisuunnikkaan erikoistapaukseksi.

Suorakulmiolle (rectangle) on olemassa useita erilaisia määritelmiä. Yleisimmän määritelmän mukaan suorakulmio on suunnikas, jossa on neljä suoraa kulmaa. Muita määritelmiä ovat muun muassa suunnikas, jossa on vähintään yksi suora kulma ja nelikulmio, jossa on neljä suoraa kulmaa. Tässä tutkimuksessa käytetään ensimmäistä määritelmää. Kaksi muuta määritelmään ovat kuitenkin ekvivalentteja ensimmäisen määritelmän kanssa.

Neljäkäs (rhombus) on määritelmän mukaan suunnikas, jonka kaikki sivut ovat yhtä pitkiä. Toisen määritelmän mukaan neljäkäs on suunnikas, jossa vähintään kaksi vierekkäistä sivua ovat yhtä pitkät. Neljäkkään vastakkaiset sivut ovat yhtä suuret ja sen vastakkaiset sivut ovat yhdensuuntaiset. Nämä ominaisuudet seuraavat suoraan määritelmästä, jonka mukaan neljäkäs on myös suunnikas. Tiedetään, että suunnikkaalle pätee edellä mainitut ominaisuudet.

Yleisimmän määritelmän mukaan neliö (square) on suorakulmio, jossa on neljä yhtenevää sivua. Muita määritelmiä ovat muun muassa suorakulmio, jonka vierekkäiset sivut ovat pareittain yhtenevät ja neliö on nelikulmio, joka on sekä neljäkäs, että suorakulmio. Nelikulmioista ainoastaan neliö voidaan määritellä usean eri nelikulmiotyypin avulla. Neliö on toisaalta myös suorakulmio, neljäkäs ja suunnikas.



Kuva 5.2: Nelikulmioiden hierarkia [34]

6 Kehittämistutkimus

Kehittämistutkimus on varsin nuori tutkimusmenetelmä opetuksen tutkimisessa. Sitä alettiin kehittää 1990-luvun alkupuolella, mutta vasta 2000-luvulla tutkimusmenetelmä on yleistynyt. Kehittämistutkimus yhdistelee kvantitatiivista ja kvalitatiivista tutkimusmenetelmää. Tutkimuksessa kehittäminen pohjautuu teoriaan ja tuottaa uutta teoriaa, mikä erottaa sen muista tutkimusmenetelmistä. Menetelmän vahvuus on se, että tutkimus tuottaa käytännöllistä tietoa, jota voidaan soveltaa opetuksessa, tutkimuksen jokaisessa vaiheessa. Toisaalta kehittämistutkimusta on menetelmän luotettavuudesta, koska menetelmälle ei ole määritelty yhteneviä tutkimuskäytäntöjä [5].

Kehittämistutkimukselle on olemassa useita erilaisia määritelmiä. Tämä tutkimus pohjautuu DiSessan ja Cobbin esittelemiin neljään teoriakategoriaan, joiden pohjalta kehittämistutkimuksella voidaan luoda uutta teoriaa. Nämä kategoriat ovat pääteoriat, ajattelua ohjaavat teoriat, toimintaa ohjaavat teoriat ja oppiainekohtaiset teoriat. [22, s.12-13].

Pääteoriat käsittelevät ilmiöitä hyvin yleisellä tasolla ja niihin perustuvan kehittämistutkimuksen haasteena on vastata yksityiskohtaisiin kehittämistarpeisiin. Toisaalta nämä teoriat pysyvät usen ajankohtaisina pitkään. Ajattelua ohjaavat teoriat tarjoavat mahdollisuuden käsitteellistää opetukseen ja oppimiseen liittyvää kehittämistä. Sen haasteena on laaja yleistys, koska tietoa saadaan rajatusta joukosta. DiSessan ja Cobbin mukaan toimintaa ohjaavat teoriat soveltuvat usein kehittämistutkimuksen teoriapohjaksi, sillä kehittämistutkimus tuottaa käytännön ratkaisuja. Toimintaa ohjaavilla teorioilla on kuitenkin niin paljon erilaisia osatekijöitä, että kaikkien huomioon ottaminen on erittäin haastavaa. Oppiainekohtaiset teoriat tarjoavat mahdollisuuden opettamiseen tarkoitettujen toimintamallien kehittämisen ja testaamisen. [22, s.12-13]

Tutkimusaineiston analysoinnissa on käytetty pitkälti laadullisen tutkimuksen metodeita. Haastatteluaineiston analyysissa on käytetty sisällönanalyysia. Aineistolähtöisellä analyysilla on pyritty luomaan teoreettinen kokonaisuus aineistosta nousevien teemojen avulla. [30] Aineistolähtöisessä analyysissä ongelmana on se, että tutkijan omat ennakkokäsitykset saattavat vaikuttaa analyysin teemoihin ja aineistoista nouseviin teemoihin. Tämä ongelma pyritään minimoimaan kirjoittamalla auki erilaiset ennakkokäsitykset, joita tutkijalla on aiheesta ja mahdollisista tutkimustuloksista.

Tässä tutkimuksessa painottuvat oppiainekohtaiset teoriat sekä toimintaa ohjaavat teoriat. Tutkimuksen tavoitteena on kehittää toimiva ratkaisu ohjelmoinnin opettamiseen peruskoulussa. Opetuksen haasteena on se, että ohjelmoinnin opetus on sisällytetty matematiikan opetukseen. Tutkimuksen tavoitteena on löytää mielekäs tapa yhdistää ohjelmoinnin ja matematiikan opetus.

7 Tutkimuskysymykset

Tutkimuksessa on käytetty seuraavia tutkimuskysymyksiä:

1. Miten menetelmä syvensi oppilaiden ymmärrystä nelikulmioiden hierarkiasta?
2. Miten oppilaat suhtautuvat ohjelmointiin osana matematiikan opetusta?

8 Tutkimuksen kulku

Tutkimuksessa haluttiin tutkia, miten ohjelmointia voitaisiin opettaa osana matematiikan optusta siten, että opetus tukisi sekä ohjelmoinnin että matematiikan oppimista. Lisäksi tutkimuksessa kartoitetaan oppilaiden asenteita tällaista opetusmuotoa kohtaan. Sekä nelikulmioiden hierarkia että graafisen ohjelmointikielen opettelu ovat osa alakoulun opetussuunnitelmaa. Tutkimuksessa päädyttiin yhdistämään graafisen ohjelmoinnin alkeet ja nelikulmioiden hierarkia. Opetuksen oli tarkoitus tukea algoritmisen ajattelun ja ohjelmointitaitojen kehittymistä sekä nelikulmioiden hierarkian ymmärtämistä. Erilaisia nelikulmioita on suhteellisen helppo piirtää ohjelmoimalla. Algoritminen ajattelu voi myös tukea erilaisten nelikulmioiden hierarkkisten erojen ymmärtämisessä. Tässä luvussa käsitellään tarkemmin aineiston laadintaa ja ratkaisuja, joiden perusteella kyseiseen aineistoon päädyttiin. Lisäksi kerrotaan opetuskokeilun ja haastattelujen kulusta.

8.1 Aineiston laadinta

Tutkimusaineiston keräämistä varten laadittiin tehtävämoniste (Liite 12.2) oppilaille. Lisäksi oppilaille laadittiin kirjallinen kyselylomake sekä laadittiin runko haastattelujen pohjaksi. Tehtävät sisälsivät helppoja, keskitasoisia ja vaikeita tehtäviä. Tehtävissä oli nelikulmioiden hierarkiaan liittyviä ohjelmointitehtäviä, algoritmiseen ajatteluun pohjautuvia ohjelmointitehtäviä, luovia tehtäviä ja kielentämistehtäviä. Tehtävät suunniteltiin niin, että oppilaat pystyivät tekemään niitä yksin tai yhdessä luokkatovereiden kanssa.

Tehtävät suunniteltiin siten, etteivät ne vaatineet oppilailta aikaisempaa ohjelmointikokemusta. Ensimmäiset tehtävät pyrittiin suunnittelemaan erittäin helpoiksi, jotta oppilaat pääsivät itsenäisesti alkuun tehtävien tekemisessä. Helpoissa tehtävissä pyrittiin tarjoamaan oppilaille mahdollisuus tutkivaan oppimiseen. Oppilaiden oli tarkoitus keksiä itse, miten ohjelmointiympäristössä voidaan tuottaa haluttu ohjelma. Tehtävien oli tarkoitus motivoida oppilaita antamalla heille kokemuksia onnistumisesta ja oivaltamisesta. Helpoissa tehtävissä pyrittiin siihen, että oppilaat tarvitsevat ratkaisun laatimiseen hieman käsitteellistä ymmärrystä ja yritteliäisyyttä. Tehtävissä pyrittiin siihen, että tehtävän ratkaisuun riittäisi hyvin yksinkertainen päättelyprosessi.

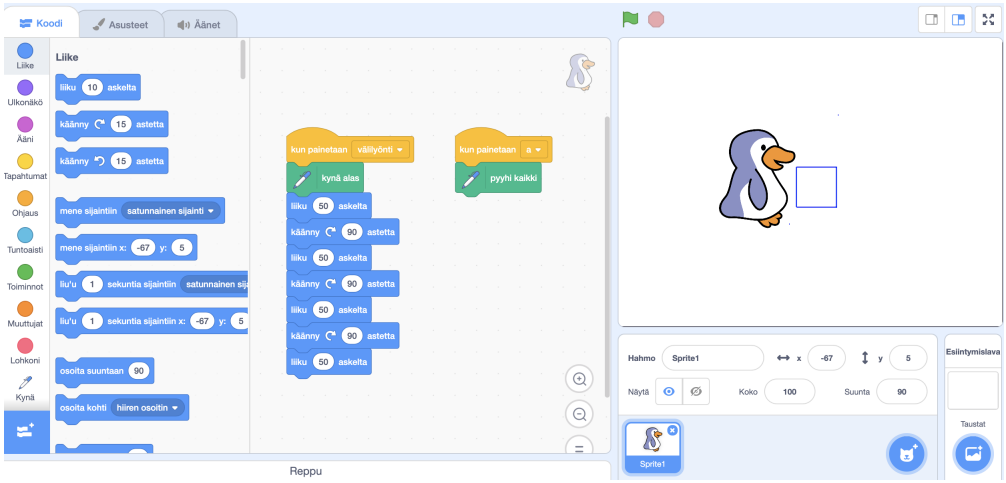
Tehtävät vaikeutuivat vähitellen ja viimeiset tehtävät oli suunniteltu erityisesti motivoituneille ja ohjelmoinnista kiinnostuneille oppilaille. Tehtävien laatimisessa oli lähtökohtana tutkiva oppimisteoria. Lisäksi vaikeammissa tehtävissä oppilailta vaadittiin etenkin luovaa ongelmanratkaisua. Tehtävät pyrittiin laatimaan siten, että oppilaat joutuvat soveltamaan oppimaansa matematiikkaa ja ohjelmointia sekä päättelemään näiden avulla ratkaisu. Tehtävissä pyrittiin haastamaan kaikkia matemaattisen osaamisen piirteitä.

Tehtäväpaperissa oli myös luovia tehtäviä, joiden tarkoitus oli motivoida oppilaita ja herättää heidän kiinnostusta ohjelmointia kohtaan. Luovien tehtävien avulla

pyrittiin luomaan kiinnostusta ja positiivista asennetta ohjelmointia kohtaan. Opetussuunnitelmassa ohjelmointitaito on sidottu vahvasti matematiikkaan. Näillä tehtävillä pyrittiin laajentamaan oppilaiden käsitystä ohjelmoinnista.

Osa tehtävistä liittyi matematiikan kielentämiseen. Oppilaita pyydettiin selittämään, miten he ovat päätyneet ratkaisuunsa. Kielentämisen tarkoituksena oli syventää oppilaiden ymmärrystä sekä kerätä tietoa oppilaiden ajatusprosesseista, joiden avulla he ovat ratkaisseet tehtävän.

Oppilaille laadittiin Scratchissä ohjelma, jota oppilaat käyttivät pohjana tehtävien tekemisessä. Ohjelmassa oli esimerkki, miten voidaan ohjelmoida neliö. Lisäksi ohjelmassa oli ominaisuus, jonka avulla pystyi poistamaan piirretyt kuviot. Malliohjelmalla oli tarkoitus varmistaa, että kaikki oppilaat pääsevät alkuun tehtävien tekemisessä. Ennen tutkimusta tutkija perusti Scratchiin opettajan tilin. Tilille lisättiin luokkahuone, jonne jaettiin malliohjelma. Lisäksi jokaiselle tutkimukseen osallistuvalle luotiin tunnukset, jotka oli linkitetty opettajan tiliin.



Kuva 8.1: Oppilaille jaettu ohjelma

Kyselylomake (12.1) pyrittiin laatimaan siten, että sillä saatiin kerättyä tietoa vastaukseksi tutkimuskysymyksiin. Lomakkeessa oli kysymyksiä, joihin vastattiin Likert asteikolla. Lisäksi oppilaat antoivat kouluarvosanan osalle tehtäväpaperin tehtävistä. Likert asteikolla vastattavissa kysymyksissä kysyttiin oppilaiden kokemusta tehtävien vaikeustasosta, mielenkiintoisuudesta ja mukavuudesta sekä ohjelmoinnin hyödyllisyydestä. Kysymyksillä pyrittiin saamaan vastauksia siihen, että onko matematiikan opettaminen ohjelmoinnin avulla mielekäs opetustapa ja miten oppilaat yleisesti kokevat ohjelmoinnin tarpeellisuuden. Tehtävien arvioinnilla pyrittiin selvittämään millaiset tehtävät oppilaat kokevat motivoivimpina ja mitkä tehtävät oppilaat kokevat epämiellyttävimpinä. Lomakkeen taustamuuttujana käytettiin ainoastaan opetettavaa luokkaa.

Oppilaiden haastatteluilla pyrittiin saamaan lisää tietoa erityisesti nelikulmioiden hierarkian ymmärtämisestä. Haastattelujen runko suunniteltiin siten, että se tuki aineistolähtöistä analyysia. Haastatteluissa Kysyttiin oppilailta erilaisista nelikulmioista. Oppilaiden tuli nimetä nelikulmiot ja kertoa niiden yhtenevyyksistä ja eroa-

vaisuuksista sekä arvioida voiko yhteen luokkaan kuuluva nelikulmio kuulua myös toiseen luokkaan. Kysymykset pyrittiin muotoilemaan siten, että oppilas sai kertoa ajatuksiaan siten, että haastattelija ei ohjaa oppilaan vastuksia. Haastateltavaksi pyrittiin saaman mahdollisimman erilaisia oppilaita, jotta haastatteluissa tulisi laajemmin esiin erilaisia ajatuksia nelikulmioista ja niiden hierarkioista.

Opettajilta pyydettiin vapaamuotoinen kirjallinen kommentti opetuksesta. Opettajilta kysyttiin oppilaiden ohjelmointikokemuksesta ja opettajien mielipiteistä opetustavasta. Palautteella pyrittiin saamaan lisää tietoa opetustavan mielekkyydestä alakoulussa. Opettajille annettiin hyvin vapaat kädet kommentointiin, sillä kommenteilla pyrittiin selvittämään asenteisiin liittyviä kysymyksiä.

8.2 Tutkimuksen kulku

Tutkimus toteutettiin kahdessa eri alakoulussa. Molemmista kouluista tutkimukseen osallistui yksi kuudes luokka. Ennen aineiston keruuta koulujen rehtoreilta ja opettajilta sekä oppilaiden vanhemmilta pyydettiin kirjallista lupaa aineison keräämiseen. Toisesta koulusta osallistui 10 oppilasta ja toisesta 15 oppilasta. Yhteensä analyysia varten saatiin 25 kyselylomaketta ja kuusi haastattelua. Lisäksi luokkien opettajilta pyydettiin kirjallinen kommentti opetuksesta. Tutkimukseen käytettiin molemmissa luokissa kaksi kaksoistuntia eli yhteensä neljä kokonaista oppituntia.

Aluksi oppilaille kerrottiin tutkimuksesta ja sen tarkoituksesta. Tämän jälkeen käytiin lyhyesti läpi Scratchin perusperiaatteita sekä jaettiin oppilaille tunnukset palveluun. Lisäksi oppilaita ohjeistettiin käyttämään Scrathin luokkahuonetta ja kopioimaan sieltä itselle malliohjelma, jonka pohjalta tehtäviä pystyi tekemään. Tähän varattiin yhteensä noin puoli tuntia.

Seuraavaksi oppilaita pyydettiin tekemään tehtävämoniste. Monisteen tekemiseen oli varattu aikaa yhteensä kolme oppituntia. Oppilaat saivat edetä tehtävissä omaan tahtiin. Useimmat oppilaat saivat kaikki nelikulmioihin liittyvät tehtävät tehtyä. Tehtävämonisteen laadinnassa päädyttiin laatimaan monen tasoisia tehtäviä, jotta tehtävissä olisi motivoivia ja haastavia tehtäviä mahdollisimman monen tasoiselle oppilaalle. Oppilaat kuitenkin suoriutuivat tehtävistä oletettua nopeammin. Oppituntien lopusta oli varattu puoli tuntia kyselylomakkeen täyttämiseen. Lomakkeen täyttämiseen varattiin runsaasti aikaa, jotta oppilaat jaksaisivat täyttää lomakkeen huolellisesti ja kiirehtimättä. Oppilaita ohjeistettiin tekemään tehtäviä yhdessä, jos ne tuntuivat haastavilta. Tarkoituksena oli ohjata oppilaat keksimään itse ratkaisut tehtäviin yksin tai yhdessä. Suurin osa oppilaista teki yhteistyötä luokkatovereiden kanssa, mutta osa keskittyi tekemään tehtäviä yksin.

Oppituntien jälkeen haastateltiin yhteensä kuutta oppilasta. Haastattelut olivat vapaaehtoisia. Oppituntien lopussa kysyttiin vapaaehtoisia haastateltavaksi ja halukkaat saivat ilmoittautua haasteltaviksi. Haastattelut toteutettiin varsinaisen oppituntien jälkeen välitunnilla tai seuraavan oppitunnin aikana. Opettajilta pyydettiin kirjallista kommentointia tutkimuksen suorittamisen jälkeen. Opettajat saivat kertoa suhteellisen vapaasti omista mielipiteistään opetuskokeilua ja ohjelmoinnin opetusta kohtaan.

8.3 Haastattelujen kulku

Haastatteluissa keskusteltiin erilaisista nelikulmioista kuvaparien avulla (Liite 12.3). Kuvaparit näytettiin oppilaille yksi kerrallaan. Aluksi oppilaita pyydettiin nimeämään erilaiset nelikulmiot. Tämän jälkeen oppilaita pyydettiin kertomaan kuvioiden eroavaisuuksista ja yhteneväisyyksistä. Oppilaat saivat vapaasti kertoa omalla tavallaan kuvioiden ominaisuuksista. Eroavaisuuksia ja yhteneväisyyksiä kysyttiin erikseen jokaisen kuvaparin kohdalla.

Sen jälkeen, kun oppilaan kanssa oli käyty kaikki kuvaparit läpi, Sen jälkeen, kun oppilaan kanssa oli käyty kaikki kuvaparit läpi, kysyttiin oppilailta nelikulmioiden hierarkiasta. Kysymyksellä pyrittiin selvittämään, millä tasolla oppilas ymmärtää kuvioiden hierarkian. Hierarkia on terminä abstrakti ja vaikea alakoululaiselle, joten aihetta lähestyttiin oppilaille tutumman arkikielen kautta. Oppilailta kysyttiin muun muassa *voiko neliö olla myös suunnikas*. Haastattelun lopuksi oppilaat saivat halutessaan kommentoida vapaasti tutkimusta ja antaa palautetta.

8.4 Tutkimusmenetelmät

Tutkimustulosten analysoinnissa yhdistettiin kvalitatiivisia ja kvantitatiivisia menetelmiä. Oppilaiden haastattelut litteroitiin ja niitä analysoitiin sisällönanalyysilla. Oppilaiden tekemät kielentämistehtävät analysointiin myös sisällönanalyysilla. Näillä menetelmillä pyrittiin löytämään teemoja, jotka toistuvat aineistossa. Haastatteluista pyrittiin löytämään erityisesti teemoja, jotka kuvasivat oppilaiden ymmärrystä erilaisten nelikulmioiden ominaisuuksista ja niiden välisistä suhteista. Teemoista pyrittiin rakentamaan teoreettinen kokonaisuus oppilaiden matemaattisesta ymmärryksestä.

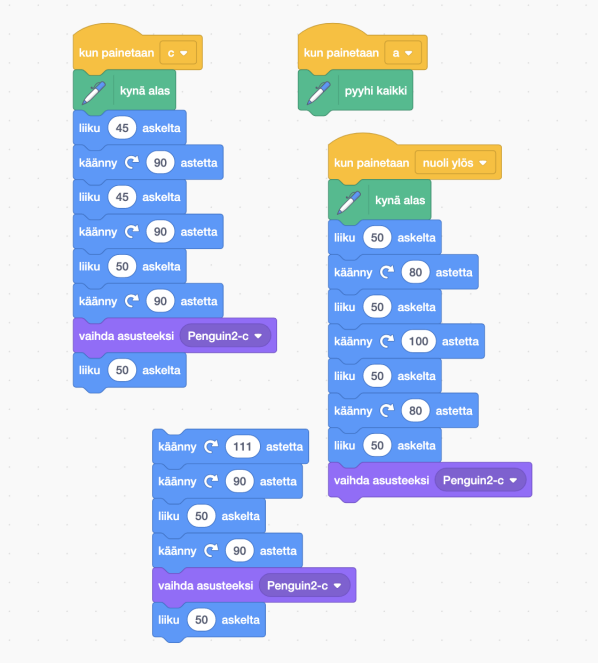
Oppilaiden täyttämien kyselyiden analysoinnissa käytettiin kvantitatiivisia menetelmiä. Kyselylomakkeen 12.1 ensimmäisen sivun tai toisen sivun kysymykset analysoitiin erikseen. Ensimmäisen sivun kysymyksille laskettiin keskiarvo, mediaani ja suhteelliset frekvenssit. Toisen sivun kysymyksille laskettiin moodi ja frekvenssit. Analyysillä pyrittiin selvittämään oppilaiden asenteita ja kokemuksia tehtävien tekemisestä ja yleisesti ohjelmointitaidon merkityksestä. Pienen tutkimusaineiston (n = 25) vuoksi analyysissä ei laskettu esimerkiksi keskihajontaa.

9 Tutkimustulokset

Tutkimustulosten mukaan oppilaat erottivat erilaiset nelikulmiot toisistaan ja he osasivat nimetä niille yhdistäviä ja erottavia tekijöitä. Sen sijaan nelikulmion hierarkian ymmärtäminen osoittautui haastavaksi. Tutkimustuloksissa nousi myös esiin suuret erot oppilaiden asenteessa ohjelmointia kohtaan luokkien välillä. Toisen luokan oppilaat kokivat ohjelmoinnin opettelun selvästi positiivisemmin ja he pitivät ohjelmointitaitoa tulevaisuuden kannalta selvästi tärkeämpänä taitona verrattuna toisen luokan oppilaisiin. Kaiken kaikkiaan oppilaat pitivät tehtäviä sopivan haastavina ja he arvioivat ohjelmointitehtävät pääosia mukaviksi tai melko mukaviksi.

9.1 Tehtävien tekeminen

Oppilaiden tehtävänä ohjelmoida graafisella ohjelmointikielellä erilaisia kuvioita. Kuvassa 9.1 on erään oppilaan keskeneräinen ratkaisu suunnikaan piirtämisestä. Ratkaisu havainnollistaa hyvin tutkivan oppimisen ajatteluprosessia, johon oppilaita pyrittiin kannustamaan. Kuvassa oppilas on tehnyt toimivat koodin suunnikkaan ja suorakulmion piirtämiseen. Suunnikasta piirtäessään oppilas on yhdistellyt ja muuttanut arvoja melko sattumanvaraisesti. Lisäksi koodista puuttuu vielä ohjauskomento, jolla kuvion voi piirtää. Oppilaita kannustettiin tällaiseen työskentelytapaan. Kokeiluvaiheen jälkeen oppilaat alkoivat hahmottaa säännönmukaisuuksia koodissa ja alkoivat hahmottaa ohjelmoinnin ja nelikulmion logiikkaa.

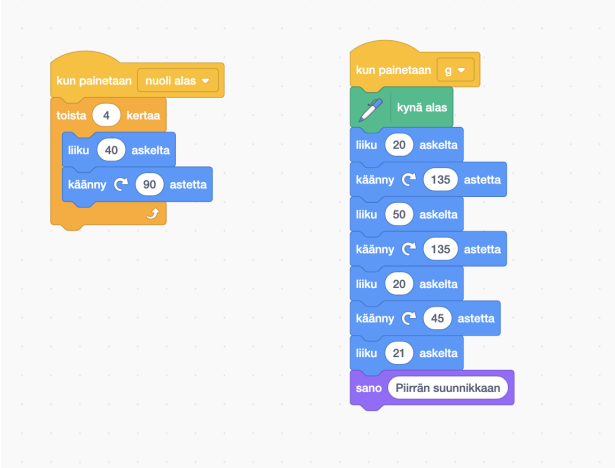


Kuva 9.1: Oppilaan keskeneräinen ratkaisu tehtävään 8 (Liite 12.2), jossa on tehtävänä piirtää suunnikas

Kuvassa 9.2 on toisen oppilaan valmis ratkaisu neliön ja suunnikkaan piirtämiseen. Neliön koodi oli annettu malliohjelmassa valmiiksi. Oppilaiden tehtävänä oli vähentää malliratkaisusta koodirivejä. Tässä tehtävässä tarvittiin algoritmista ajattelua ja matematiikan käsitteellistä ymmärrystä, sillä neliön muodosta oli löydettävä aluksi säännönmukaisuus, joka on yleistettävissä kaikkiin neliöihin. Suunnikkaan piirtämisessä oppilaan piti hahmottaa suunnikkaan säännönmukaisuudet ja niiden avulla kehittää oikeanlainen koodi.

Oppilailla oli tehtävänä ohjelmoinnin lisäksi kielentää ratkaisut erilaisten nelikulmioiden piirtämiseen. Oppilaiden kielentämistehtävien vastaukset voidaan jakaa kolmeen eri ryhmään. Ensimmäisen ryhmän vastauksissa oppilas tyypillisesti kirjoitti tekemänsä koodin paperille ja tällä tavoin kertoi, miten hän oli päätynyt ratkaisuunsa. Toisen ryhmän vastauksissa oppilaat selittivät omin sanoin, miten he olivat ratkaisuun päätyneet. Kolmannen ryhmän vastaukset, eivät anna juurikaan informaatiota oppilaan ratkaisusta. Näissä vastauksissa vastaus oli jätetty tyhjäksi tai vastaus ei kertonut mitään olennaista oppilaan ajatusprosessista. Useissa tehtäväpapereissa esiintyi useamman tyyppisiä ratkaisuja. Suurimmaksi osaksi ensimmäisen ryhmän vastauksia esiintyi yhdeksässä tehtävämonisteessa, toisen ryhmän vastauksia esiintyi eniten kymmenessä monisteessa ja kolmannen ryhmän vastauksia kuudessa monisteessa.

Ensimmäisessä ryhmässä oppilaat selittivät yksityiskohtaisesti, kuinka he olivat päätyneet ratkaisuunsa. Esimerkiksi eräs oppilas vastasi kysymykseen 3, jossa kysyttiin suorakulmion piirtämisestä, seuraavasti ”kun painetaan välilyönti niin liikkuu, käänny 90, liiku 70, käänny 90, liiku 50, käänny 90, liiku 70”. Osa oppilaista teki teh-



Kuva 9.2: Oppilailaan valmis ratkaisu neliön ja suunnikaan piirtämiseen (tehtävät 8 ja 9)

täviä mieluummin englannin kielisellä Scratchillä. Tämä näkyy myös vastauksissa. Yhdessä tehtävämonisteessa on kuvattu neljäkkään piirtämistä seuraavasti ”vaihdoin ”turn 90 degrees”:sseistä kaksi 100, ja yhden 80”. Oppilaat, jotka olivat pääasiassa vastanneet ensimmäisen ryhmän tyyppisiä vastauksia, osasivat tehdä tehtävät ja heistä kaikki olivat ehtineen oppitunneilla tekemään kaikki kielentämiseen liittyvät tehtävät.

Toisen ryhmän vastauksissa vastaukset oli esitetty pääosin hyvin lyhyesti, mutta toisaalta vastauksista käy ilmi olennaisimmat muutokset. Esimerkiksi yksi oppilas kertoi piirtäneensä suunnikkaan näin: ”vaihdoin kävelypituuden samaksi ja vaihdoin kääntymisasteita”. Moni oppilas kertoi vastauksissaan ohjelmaan tekemistään muutoksista vielä lyhyemmin, kuten ”muutin kaikki sivut saman pituisiksi”. Toisen ryhmän vastauksissa oli havaittavissa se, että osalla oppilaista oli vaikeuksia kielentää ratkaisuja. Esimerkiksi yksi oppilas kuvaa ratkaisuaan seuraavasti, ”vaihdoin askelien määrää kääntyessä” tehtävässä 6, jossa kysytään ratkaisua neljäkkään piirtämiseen. Ratkaisussa ohjelmaa piti muuttaa siten, että vastakkaisten kulmien suuruuksia muutetaan.

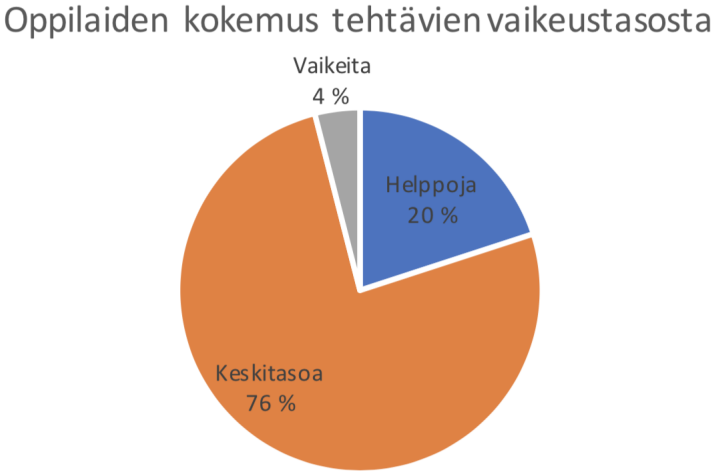
Kolmanteen ryhmään kuuluvia vastauksia oli selvästi vähiten. Näistä vastauksista ei käy ilmi, miten oppilas ratkaisi tehtävän tai onnistuiko hän sen ratkaisemaan. Vastaustyyli olivat pääasiassa epäinformatiivinen kuten ”liikutin pingviiniä” tai ”painelin nappeja”. Tutkimusaineiston pohjalta ei voida löytää selitystä näille vastauksille. Oppilailla on mahdollisesti ollut vaikeuksia kielentää vastaustaan tai hän ei ole jostain syystä halunnut antaa informatiivisempaa vastausta.

9.2 Kyselylomakkeen tulokset

Kaikki tutkimukseen osallistuvat oppilaat täyttivät kyselylomakkeen, joka on esitetty liitteessä 12.1. Tuloksia analysoidessa luokkien tulokset on yhdistetty (n = 25).

Lisäksi tuloksissa on yhdistetty arvosanat 1-6, 7-8 ja 9-10 analysoinnin helpottamiseksi.

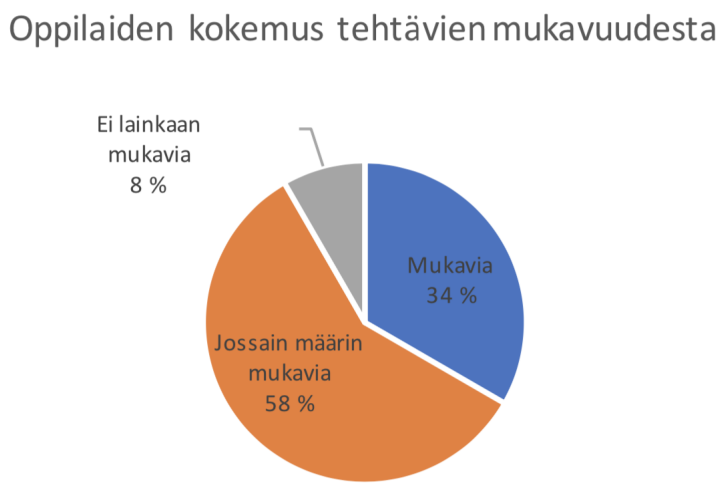
Lomakkeen ensimmäisellä kysymyksellä haluttiin selvittää, pitivätkö oppilaat tehtäviä helppoina vai vaikeina. Vastausvaihtoehdoissa neljä tarkoitti, että tehtävät olivat erityisen vaikeita ja kymmenen, että tehtävät olivat erittäin helppoja. Pääsääntöisesti oppilaat pitivät vaikeustasoa keskitasoisena. Yleisin arvosana ja mediaani olivat molemmat kahdeksan. Kaikkien vastausten keskiarvo oli 7,9. Muutama oppilas koki tehtävät helpoiksi ja vain yksi koki ne vaikeiksi. (Kuva 9.3) Näiden vastausten valossa voidaan päätellä, että tehtävät olivat tasoltaan sopivia kuudesluokkalaisille. Oppilaista suurin osa ehti tehdä kaikki tehtävämonisteen tehtävät, joista osa oli suunniteltu vaikeustasoltaan haastaviksi. Siitä huolimatta oppilaat arvioivat tehtävien vaikeustason keskitasoiseksi. Tämä saattaa kertoa siitä, että oppilaat ovat tottuneet käyttämään tietokonetta ja hyödyntämään algoritmista ajattelua, minkä takia he eivät kokeneet ohjelmointia erityisen vaikeaksi.



Kuva 9.3: Oppilaiden (n = 25) kokemus tehtävien vaikeustasosta.

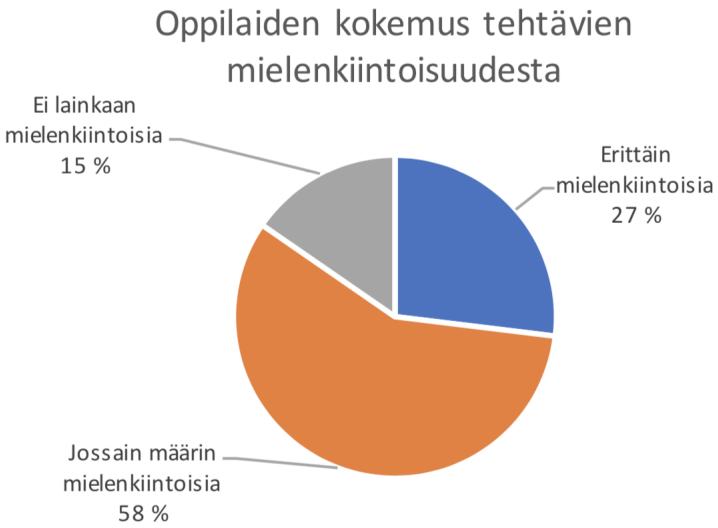
Lomakkeen toisessa kysymyksessä kysyttiin, pitivätkö oppilaat tehtävistä 9.4. Oppilaiden piti arvioida mihin heidän kokemuksensa sijoittuu asteikolla, jossa kymmenen tarkoittaa tehtävien olleen erittäin mukavia ja neljä erittäin ahdistavia. Pääosin oppilaat pitivät tehtäviä jossain määrin mukavina. Vastausten mediaani ja keskiarvo olivat molemmat 8. Kahta vastausta lukuun ottamatta oppilaat kokivat tehtävät jossain määrin mukavina tai mukavina. Matematiikan ja ohjelmoinnin yhdistäminen oppitunnilla koettiin siis pääosin vähintään melko positiivisesti. Tulokset kannustavat kehittämään matematiikkaa ja ohjelmointia yhdistäviä tehtäviä jatkossakin alakoulun opetuksessa.

Oppilailta kysyttiin myös tehtävien mielenkiintoisuudesta (Kuva 9.5). Lomakkeen asteikolla neljä tarkoittaa, että tehtävät olivat erittäin tylsiä ja kymmenen, että ne ovat erittäin mielenkiintoisia. Tässä oppilaiden kesken tuli suhteellisen paljon hajontaa. Suurin osa oppilaista koki tehtävät melko mielenkiintoisiksi tai mielenkiintoi-



Kuva 9.4: Oppilaiden (n = 25) kokemus tehtävien mukavuudesta.

siksi, mutta joukossa oli myös jonkin verran oppilaita, jotka kokivat tehtävät tylsiksi. Vastausten mediaani oli 8 ja keskiarvo 7,68. Tuloksissa korostui myös luokkien välinen ero, sillä ensimmäisen luokan keskiarvo oli 8,7 ja toisen 7. Oppilaiden kokemus heijastelee oppilaiden asenteita, joita myös kysyttiin lomakkeella. Luokka, joka koki ohjelmointitaidot hyödylliseksi taidoksi, koki myös tehtävät mielenkiintoisemmiksi.

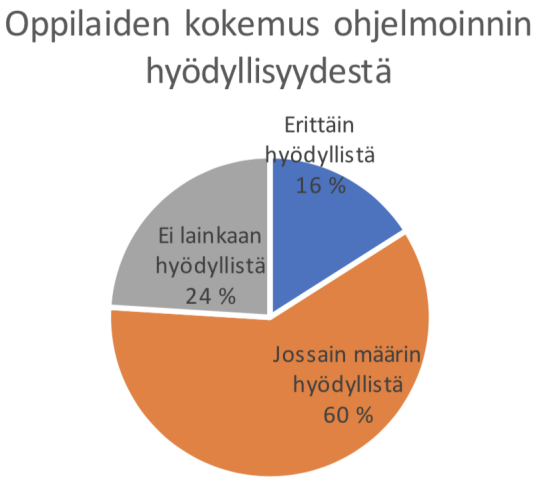


Kuva 9.5: Oppilaiden (n = 25) antamat arvosanat tehtävien mielenkiintoisuudesta

Viimeisenä kysymyksenä kysyttiin oppilaiden kokemusta ohjelmoinnin hyödyllisyydestä (Kuva 9.6). Kysymykseen vastattiin asteikolla, jossa neljä tarkoittaa ohjelmointitaidon olevan tarpeeton taito ja kymmenen hyödyllinen taito. Kysymys jakoi oppilaiden mielipiteet ja oppilaat antoivat suhteellisen tasaisesti kaikkia asteikon arvosanoja. Vastausten keskiarvo oli 7,12 ja mediaani 7. Vastauksissa näkyi selvä ero

luokkien välillä. Ensimmäisen luokan keskiarvo oli 8 ja toisen luokan keskiarvo oli 6,53. Tulokset herättävät kysymyksen suurista asenne-eroista luokkien välillä. Tämä tutkimus ei kuitenkaan tarjoa selkeitä vastauksia asenne-erojen syihin.

Tutkimuksen yhteydessä luokan omalta opettajalta kysyttiin luokan taustoja ohjelmoinnin suhteen. Toisessa luokassa opettaja oli kertonut oppilaille ohjelmoinnista ja osa oppilaista oli ohjelmoinut aikaisemmin graafisella ohjelmointikielellä. Oppilailla oli aikaisemmin ollut mahdollisuus osallistua myös koodauskerhoon. Toisen luokan oppilailla oli vähemmän kokemusta ohjelmoinnista eikä ohjelmointia oltu opetettu koulussa aikaisemmin. Eroja saattaa selittää myös se, että tässä koulussa ei ollut riittävästi tietokoneita kaikille oppilaille. Nämä erot saattavat selittää kyselytutkimuksessa esiin tulleita asenne-eroja.

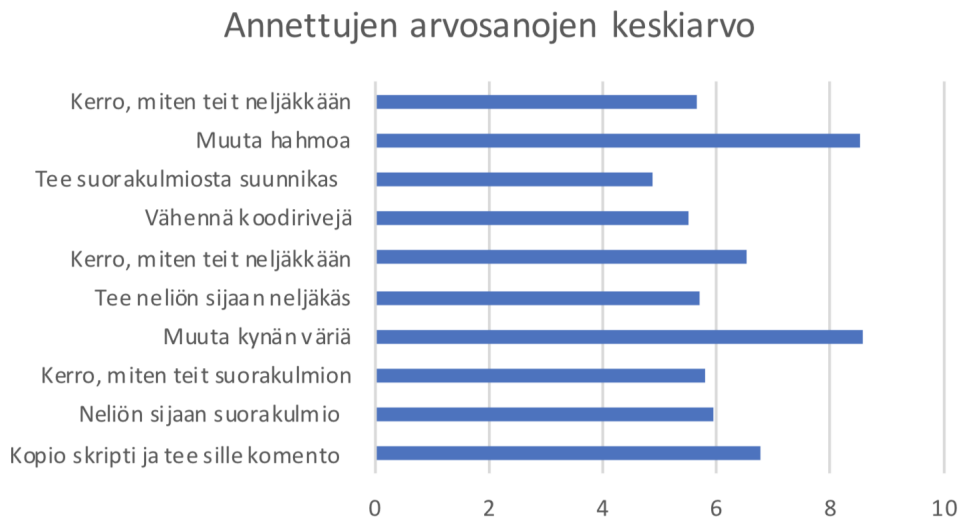


Kuva 9.6: Oppilaiden (n = 25) kokemus ohjelmointitaidon hyödyllisyydestä.

Oppilaita pyydettiin arvioimaan osaa tehtävistä asteikolla 4-10 sen mukaan, mistä tehtävästä oppilas piti eniten (Liite 12.1). Oppilaat antoivat arvosanan neljä sille tehtävälle, josta he pitivät vähiten ja arvosanan 10 sille tehtävälle, joka heidän mielestään tuntui mukavimmalta. Oppilaiden vastuksissa (Kuva 9.7) korostui selvästi se, että oppilaat pitivät eniten luovuutta vaativista tehtävistä. Näissä tehtävissä oppilaiden tuli hyödyntää algoritmista ajattelua, mutta aiheena olleet nelikulmiot sivuutettiin. Tulos kannustaa opettamaan algoritmista ajattelua tutkivalla ja luovalla asenteella.

Oppilaat kokivat epämiellyttävimpinä tehtävinä kielentämiseen liittyvät tehtävät, joissa oppilaiden piti sanallistaa tekemänsä matemaattiset tehtävät. Tähän tulokseen saattaa vaikuttaa se, että nämä tehtävät tehtiin tietokoneen sijaan kynällä ja paperilla. Tulokseen voi vaikuttaa myös muut tekijät kuten oppilaiden tottumus kielentämiseen, mutta näitä muuttujia en otettu tässä tutkimuksessa huomioon.

Nelikulmioiden ohjelmointiin liittyvät tehtävät oppilaat arvioivat keskitasolle. Näitä tehtäviä ei koettu erityisen mukaviksi, mutta toisaalta niitä ei myöskään koettu erityisen epämukaviksi.



Kuva 9.7: Oppilaiden (n = 25) antamien arvosanojen keskiarvo tehtäville (12.1). Oppilaat antoivat arvosanoja välillä 4-10, missä 10 tarkoittaa mukavinta tehtävää ja 4 epämukavinta tehtävää.

9.3 Haastattelujen tulokset

Haastatteluista nousi esille se, että oppilaat osasivat pääosin nimetä erilaiset nelikulmiot sekä löysivät niiden väliltä eroavaisuuksia ja yhteneväisyyksiä. Toisaalta nelikulmioiden välisen hierarkian ymmärtäminen osoittautui erittäin haastavaksi. Tutkimuksessa haastateltiin kuutta oppilasta. Kaikki haastateltavat osasivat nimetä ainakin osan nelikulmioista sekä kertoa niiden välisistä eroista. Parilla oppilaalla oli vaikeuksia nimetä yhteneväisyyksiä kaikki ja vain yksi haastateltava ymmärsi nelikulmioiden välisen hierarkian. Osa oppilaista oli hierarkian suhteen epävarmoja ja osa piti hierarkiaa mahdottomana.

Oppilaille näytettiin erilaisia kuvapareja (Liite 12.3). Aluksi heitä pyydettiin nimeämään erilaiset nelikulmiot. Tämä onnistui oppilailta pääosin hyvin. Tämän jälkeen oppilaita pyydettiin nimeämään kuvaparista yhtenevyyksiä ja eroavaisuuksia. Moni oppilas osasi nimetä olennaisimmat yhtenevyydet ja erot. Suurin osa oppilaista ei kuitenkaan osannut käyttää matemaattisia termejä nimeämiseen. Esimerkiksi suurin osa oppilaista kuvaili teräviä ja tylppiä kulmia vinoiksi kulmiksi.

Oppilaiden nimeämistä yhtäläisyyksistä nousivat esiin erityisesti sivujen pituudet ja kulmien lukumäärä. Lisäksi muutamassa vastauksessa oppilaan mainitsivat sivujen yhdensuuntaisuuden yhtenevyytenä. Oppilaiden mainitsemat yhtäläisyydet on esitetty kootusti taulukossa 9.1. Siinä on jaettu oppilaiden vastaukset kategorioihin teemojen mukaan. Erilaiset hierarkialuokat määritellään sivujen pituuksien ja yhdensuuntaisuuden sekä kulmien suuruuden avulla. Moni oppilas löysi näitä yhtenevyyksiä, mikä viittaa oppilaiden ymmärtävän melko hyvin erilaisten nelikulmioiden määritelmät. Kulmien suuruus mainittiin yhdistävänä tekijänä kuitenkin vain kahdesti, vaikka kulman suuruus on määrittävä tekijä kaikissa hierarkialuokissa. Kulman

suuruus osattiin kuitenkin nimetä usein erottavana tekijänä. Oppilailla oli hyvin erilaisia tapoja kuvata kuvioiden ominaisuuksia. Analyysissa nämä kuvaukset jaettiin aineistosta nousevien teemojen alle.

Taulukko 9.1: Taulukossa on koottuna oppilaiden (n = 25) nimeämiä yhtäläisyyksiä erilaisten nelikulmioiden välillä

Yhteneväisyydet	Lukumäärä
Sivun pituus	6
Kulmien lukumäärä	5
Sivujen yhdensuuntaisuus	3
Kulman suuruus	2
Muut	2

Kysyttäessä nelikulmioiden eroavaisuuksista, oppilaat mainitsivat useimmin erot sivujen pituuksissa ja kulmien suuruudessa. Erottavana tekijänä mainittiin sivujen yhdensuuntaisuus vain kerran. Muita eroja oppilaat mainitsivat vain vähän. Hierarkialuokat erotetaan toisistaan pääsääntöisesti kulmien suuruuksien, sivujen pituuksien ja niiden yhdensuuntaisuuden avulla. Tulos vahvistaa käsitystä siitä, että oppilaat ovat sisäistäneet hierarkialuokkien määritelmät hyvin. Taulukossa 9.2 on esitetty yhteenveto haastatteluissa oppilaiden mainitsemista eroavaisuuksista.

Taulukko 9.2: Taulukossa on koottuna oppilaiden (n = 25) nimeämiä eroavaisuuksia erilaisten nelikulmioiden välillä

Eroavaisuudet	Lukumäärä
Sivun pituus	11
Kulman suuruus	6
Muut	3
Sivujen yhdensuuntaisuus	1

Oppilaat osasivat luokitella erilaiset nelikulmiot ja kertoa niiden ominaisuuksista. Sen sijaan heillä oli vaikeuksia nähdä hierarkkinen rakenne luokkien välillä. Oppilailta kysyttiin, voiko neliö olla myös suunnikas. Suurin osa haastateltavista ei osannut vastata kysymykseen oikein tai he eivät osanneet perustella vastaustaan. Haastateltavista yksi osasi vastata kysymykseen oikein ja perustella sen. Haastateltavista kolme ei osannut vastata kysymykseen. He olivat epävarmoja vastauksesta, koska he eivät osanneet perustella sitä. Kaksi vastasi, että neliö ei voi olla suunnikas. Molemmat vastaajat perustelivat vastauksen sillä, että suunnikkaassa kulmat eivät voi olla 90 °.

Haastatteluista voidaan päätellä, että ohjelmoinnillinen tapa opetella nelikulmioiden hierarkiaa ei juurikaan syventänyt oppilaiden ymmärrystä hierarkiasta. Sen sijaan ohjelmoinnillinen oppiminen näyttäisi auttavan oppilaita huomaamaan erilaisten nelikulmioluokkien eroja ja samanlaisuuksia. Toisaalta tässä tutkimuksessa ei

otettu huomioon oppilaiden aikaisempaa osaamista ja sen vaikutusta tehtävien tekemiseen. On mahdollista, että oppilaille nelikulmioiden luokittelu oli jo entuudestaan tuttu asia. Kaiken kaikkiaan oppilaat osasivat pääsääntöisesti nimetä nelikulmiot ja ymmärsivät hierarkialuokkien määritelmät, mutta luokkien välisen yhteyden ymmärtäminen oli haastavaa. Lisäksi oppilaat eivät pääsääntöisesti osanneet kuvailla havaintojaan matemaattisin termein, vaan käyttivät tutumpaa arkikieltä.

9.3.1 Matemaattinen ilmaiseminen

Haastatteluiden perusteella oppilaiden tapa ilmaista matemaattisia eroavaisuuksia erilaisten kuvioiden välillä voidaan jakaa kolmeen tyyppiin. Ensimmäisessä tyypissä oppilaat keksivät itse sanoja korvaamaan matemaattisen sanaston puutetta. Oppilaat puhuivat esimerkiksi suunnikkaan lattiasta ja katosta kuvatessaan nelikulmion sivuja tai he kuvasivat nelikulmiota, jonka kulmat eivät ole yhdeksänkymmentä astetta, koukistuneiksi tai vinoiksi. Toisessa tyypissä kuvailtiin nelikulmiota käyttäen apunaan toiseen luokkaan kuuluvaa nelikulmiota. Esimerkiksi suorakulmiota he kuvasivat kahden neliön yhdistelmäksi. Kolmannessa tyypissä käytettiin matematiikasta puhuttaessa eksakteja matemaattisia termejä kuten kulman suuruus tai sivun pituus. Moni oppilas kuvaili eroja käyttäen vastauksessaan monen tyyppisiä kuvailuja.

Ensimmäinen tyyppi oli selvästi yleisin tapa kuvailla nelikulmioiden ominaisuuksia haastatteluissa. Etenkin suunnikkaan kulmia lähes jokainen oppilas kuvaili vinoksi. Lisäksi usea oppilas kuvaili suurempaa sivunpituutta sanomalla kuvion olevan isompi kuin toinen kuvio. Toisaalta oppilaat, jotka käyttivät arkikieltä kuvioiden omaisuuksien kuvailuun, pääsääntöisesti löysivät oleellisimmat erot ja yhteneväisyydet kuvioiden väliltä.

Toista tyyppiä esiintyi haastattelussa huomattavasti vähemmän kuin ensimmäistä tyyppiä, mutta useampi oppilas käytti tätä tapaa kuvaillessaan nelikulmioita. Haastatteluissa yksi kuvapari esitti neliötä ja suorakulmiota. Oppilaan kuvailivat suorakulmiota esimerkiksi seuraavilla tavoilla; *kaks neliöä ja sitte ne on yhistetty* ja *niinku se ensimmäinen että se on niinku kaks tollasta*. Nämä oppilaat osasivat nimetä erilaiset nelikulmiot, mutta heillä oli enemmän vaikeuksia hahmottaa erilaisten hierarkialuokkien eroja. Tämän tyyppisissä vastauksissa oppilaat huomasivat, että nelikulmioiden välillä on eroja, mutta he eivät osanneet nimetä ominaisuuksia, jotka erottivat ja yhdistivät nelikulmioita. Kaikki oppilaat, joilla oli vastauksissaan tämän tyyppisiä kuvailuja, vastasivat kieltävästi kysymykseen *voiko neliö olla myös suunnikas*. Perusteluksi he kertoivat, ettei suunnikkaassa voi olla suoria kulmia.

Kolmannen tyypin vastauksia oli jonkin verran haastatteluissa, mutta niitä oli selvästi vähemmän kuin ensimmäisen tyypin vastauksia. Tämän tyyppisissä vastauksissa oppilaat osasivat käyttää eksakteja matemaattisia termejä kuvaillessaan eroavaisuuksia ja yhtenevyyksiä. Yleisimmin oppilaat osasivat puhua suorista kulmista ja sivun pituudesta matemaattisin termein. Useat oppilaat yhdistelivät vastauksissaan tämän tyypin ja ensimmäisen tyypin kuvailuja nelikulmioiden ominaisuuksista. Haastatteluissa yksi oppilas ymmärsi nelikulmioiden hierarkian. Tämä oppilas osasi käyttää eksakteja matemaattisia termejä vastauksissaan selvästi enemmän verrattuna muihin haastateltaviin. Moni oppilas, jonka vastauksissa oli kolmannen tyypin

vastauksia, epäröi kysymyksessä voiko nelikulmio olla myös suunnikas. Nämä oppilaat vastasivat useimmin *ehkä* tai *en osaa sanoa*, mutta eivät osanneet perustella vastaustaan.

Tulosten perusteella matemaattisen ilmaisemisen tasolla oli yhteys matematiikan ymmärtämiseen. Oppilaat, jotka osasivat käyttää puheessaan matemaattisia termejä sujuvasti, osasivat nimetä olennaisimmat erot ja yhtenevyydet kuvioiden väliltä varattuna haastateltaviin, jotka käyttivät puheessaan vain arkikieltä tai arkikieltä ja muita nelikulmioluokkia. Lisäksi he useimmin ymmärsivät hierarkian tai pitivät sen olemassaoloa mahdollisena. Arkikielen käyttäminen kuvailussa oli myös erittäin yleistä, mikä saattaa kertoa siitä, että matemaattisia termejä ei olla totuttu käyttämään matematiikasta puhuessa. Toisaalta oppilaalla saattaa olla hyvä käsitteellinen ymmärrys, vaikka hän ei osaisi tuoda sitä verbaalisesti esille. Niillä oppilailla, jotka eivät hallinneet matemaattista sanastoa tai käyttivät vastauksissaan nelikulmioiden luokkia arkikielen sijaan, oli selvästi enemmän vaikeuksia ymmärtää erilaisten nelikulmioiden määritelmiä ja määritelmien eroja. Oppilaille, joilla oli vaikeuksia matemaattisten termien kanssa, käsiteellisen ymmärryksen taso oli heikompi kuin niillä oppilailla, jotka hallitsivat sanaston paremmin.

10 Johtopäätökset

Tutkimustulosten perusteella oppilailla oli selvästi vaikeuksia ymmärtää nelikulmioiden hierarkiaa. Oppilaat osasivat nimetä melko hyvin erilaiset nelikulmiot ja niiden ominaisuudet, mutta hierarkian ymmärtäminen tuotti vaikeuksia. Oppilaat kuitenkin suhtautuivat matematiikan opiskeluun ohjelmoinnin avulla pääosin positiivisesti. Tehtäviä ei juurikaan koettu erityisen vaikeiksi tai ahdistaviksi, vaikka mukana oli tehtäviä helpoista erittäin vaikeisiin. Opettajilla ei tällä hetkellä ole juurikaan käytössä ohjelmointia ja matematiikkaa yhdistäviä materiaaleja, joita he voisivat hyödyntää opetuksen suunnittelussa. Tutkimuksen perusteella vastaavaa opetustapaa olisi hyödyllistä kehittää toimivammaksi ja opettajille suunnattuja matematiikkaa ja ohjelmointia yhdisteleviä opetusmateriaaleja olisi syytä kehittää tuntien suunnittelun tueksi.

10.1 Nelikulmioiden hierarkian ymmärrys

Tuloksissa nousi esiin oppilaiden haaste ymmärtää nelikulmioiden hierarkiaa. Pääsääntöisesti oppilaat huomasivat erilaisten nelikulmioiden väliset erot sekä yhteneväisyyden. Sen sijaan oppilailla oli vaikeuksia ymmärtää nelikulmioiden välistä hierarkkista rakennetta. Tulosten perusteella nelikulmioiden opetus ohjelmoinnin avulla ei juurikaan auttanut oppilaita syventämään ymmärrystä nelikulmioiden hierarkiasta. Tutkimuksessa ei otettu huomioon oppilaiden lähtötasoa, joten vertailua oppilaiden ymmärryksestä ennen ja jälkeen opetuksen ei voitu tehdä. Toisaalta hierarkia on käsitteenä hyvin abstrakti ja siksi sen ymmärtäminen voi olla haastavaa alakouluikäiselle.

Oppilaiden kyky löytää erottavat ja yhdistävät ominaisuudet kertovat etenkin oppilaiden käsitteellisen ymmärtäminen tasosta, joka on tärkeä osa-alue matemaattisessa ongelmanratkaisussa. Käsitteellisen ymmärtämisen tason kehittymistä voisi tukea alakoulussa etenkin kielentämisen avulla. Kaikilla haastateltavilla oli ainakin jossain määrin haasteita tuoda verbaalisesti esiin ymmärrystään, vaikka he olisivatkin ymmärtäneet asian. Tämä viittaa siihen, että matematiikan tunneilla ei välttämättä harjoitella matematiikan ilmaisemista sanallisesti käyttäen matemaattista kieltä tai sen harjoittelu ei ole säännöllistä. Tutkimuksessa kävi ilmi, että matemaattisen ilmaisemisen tasolla tai käsitteellisen ymmärryksen tasolla oli selvästi yhteys, mikä tukee kielentämisen merkitystä matematiikan osaamisen kehittymisessä.

Abstraktien käsitteiden lisäksi kyselylomakkeen tuloksista kävi ilmi, että oppilaat pitivät kielentämiseen liittyvistä tehtävistä vähiten. Tutkimustuloksista ei kuitenkaan käy ilmi havainnon syytä. Tulos voi kertoa siitä, että oppilaat eivät ole tottuneet kielentämiseen tai se tuntui vaikealta. Tulokseen voi toisaalta vaikuttaa myös se, että muut tehtävämonisteen tehtävät koettiin miellyttävämmäksi verrattuna kielentämistehtäviin ja se, että kielentäminen tapahtui tietokoneen sijaan kynällä ja paperilla. Kaiken kaikkiaan oppilaat pitivät miellyttävimpinä luovia tehtäviä, jotka eivät suoraan liittyneet matematiikkaan.

10.2 Ohjelmointi osana matematiikan opetusta

Oppilaat kokivat matematiikan opiskelun ohjelmoinnin avulla pääosin positiivisesti. Tulosten mukaan selvästi suurin osa oppilaista koki tehtävät mukaviksi tai melko mukaviksi. Lisäksi suurin osa koki tehtävät mielenkiintoisiksi tai melko mielenkiintoisiksi. Kaiken kaikkiaan oppilaat suhtautuivat matematiikan ohjelmoinnilliseen opetukseen joko melko positiivisesti tai positiivisesti. Oppilaiden kokemusten pohjalta opetustapa voisi olla toimiva lisä perinteiselle tavalle opettaa matematiikkaa.

Kyselylomakkeen tuloksista nousi esiin yksi yllättävä seikka. Oppilaiden mielipiteet ohjelmointitaidon hyödyllisyydestä jakoivat luokat erittäin vahvasti kahtia. Toisen tutkittavan luokan oppilaat kokivat, että ohjelmointitaidoista on hyötyä tulevaisuudessa ja toisen luokan oppilaat kokivat ohjelmoinnin melko hyödyttömänä taitona. Tutkimustuloksista ei kuitenkaan löydy syytä tähän tulokseen. Oppilaiden mielipiteeseen saattaa kuitenkin vaikuttaa opettajan tietoisuus ohjelmoinnista ja sen hyödyntämisestä. Opiskelumotivaatioon voi vaikuttaa olennaisesti myös käytössä olevat välineet. Graafinen ohjelmointi vaatii riittävän hyvän tietokoneet, jotta ohjelmointi sujuu ongelmitta. Luokan, joka koki ohjelmointitaidon hyödyllisenä, opettaja oli kertonut ohjelmoinnista ja sen käytöstä tietoteknisissä laitteissa ja lisäksi oppilailla oli käytössä tietokoneluokka. Lisäksi oppilailla oli ollut mahdollisuus osallistua ohjelmoinnin opetukseen aikaisemmin. Ohjelmointitaitoon negatiivisemmin suhtautuneilla oppilailla ei ollut käytössä tietokoneita kaikille oppilaille eikä heidän opettajallaan ollut juurikaan tietoa ja kokemusta ohjelmoinnista tai sen opettamisesta. Tämä saattaa osittain selittää luokkien välisiä eroja, mutta aiheesta ei ole juurikaan löydettävissä tutkimustietoa johtopäätösten tueksi.

10.3 Harjoitusten kehittäminen ja hyödyntäminen

Tutkimustulosten perusteella ohjelmointia voidaan käyttää lisänä matematiikan opetuksessa. Oppilaan suhtautuivat opetustapaan pääosin positiivisesti ja monet kokivat ohjelmoinnin hyödylliseksi taidoksi. Toisaalta opetus ei vaikuttanut syventävän matemaattista ymmärrystä erityisen hyvin. Haastatteluiden perusteella oppilailla oli jonkin verran käsitteellistä ymmärrystä, mutta opetuksen tuloksena he eivät oppineet nelikulmioiden hierarkkista rakennetta. Oppilaiden positiivinen asenne voi kutenkin vaikuttaa positiivisesti matematiikan oppimiseen ja opiskeluun. Ohjelmointitehtäviä on kuitenkin tarpeen kehittää tukemaan matematiikan oppimista. Ohjelmointitehtäviä olisi syytä kehittää siten, että ne tukisivat ja syventäisivät matematiikan oppimista.

Tehtävämonisteeseen oli suunniteltu helppoja, keskitasoisia ja vaikeita tehtäviä. Lisäksi oli tehtäviä, joiden oli tarkoitus motivoida oppilaita ja kannustaa heitä luovuuteen matemaattisten tehtävien lomassa. Helpot ohjelmointitehtävät oli suunniteltu siten, että oppilas kykenee mallin avulla tekemään halutun ohjelman siten, että ohjelmaan ei tarvitse tehdä suuria muutoksia. Esimerkiksi (Liite 12.2) tehtävä 2 on suunniteltu siten, että oppilaan on muutettava ohjelmassa vain sivun pituuksia, mutta ohjelmaan ei tarvitse lisätä eikä poistaa toiminnallisuutta. Nämä tehtävät sujuivat op-

pilailta hyvin ja he saivat ne pääosin nopeasti tehtyä. Lisäksi kielentämistehtävät oli kategorisoitu helpoiksi tehtäviksi, sillä niissä oppilaan tuli kertoa kuinka hän oli päätynyt edellisen kohdan ratkaisuun. Teoreettisesta näkökulmasta helpoissa tehtävissä tarvittiin hieman käsitteellistä ymmärrystä ja yritteliäisyyttä, mutta proseduraalista sujuvuutta, mukautuvaa päättelyä ei vaadittu juuri lainkaan.

Tehtävämonisteeseen olisi tarpeellista lisätä etenkin keskitason tehtäviä. Keskitasoiset tehtävät oli suunniteltu siten, että käsitteellisen ymmärryksen ja yrittelijäisyyden lisäksi ratkaisu vaati myös proseduraalista sujuvuutta ja päättelyä. Proseduraalinen sujuvuus yhdistettiin tehtävien laadinnassa etenkin algoritmiseen ajatteluun. Esimerkiksi tehtävämonisteen 12.2 tehtävä 11 oli keskitasoinen. Siinä oppilaan tuli vaihtaa useita skriptejä ja muuttaa kulmien pituuksia sekä päätellä skriptien järjestys. Näitä tehtäviä olisi voinut olla tehtävämonisteessa enemmän. Näiden tehtävien oli tarkoitus kehittää matemaattisen osaamisen lisäksi algoritmista ajattelua siten, että vaativuustaso sopii oppilaille, joilla ei ole juurikaan aikaisempaa ohjelmointikokemusta. Lisäksi keskitason tehtävissä ei tarvittu matemaattista osaamista, joka ei liittynyt tunnin aiheeseen.

Monisteessa vaikeat tehtävät oli suunniteltu haastamaan etenkin ohjelmoinnista kiinnostuneita ja motivoituneita oppilaita. Esimerkiksi monisteen tehtävä 16 *Muuta neliön skriptiä niin, että neliön sivun pituus on sattumanvarainen luku väliltä 20-100*. Funktiot ja muuttujat eivät kuulu alakoulun matematiikan tavoitteisiin, mutta ohjelmoinnista kiinnostuneet oppilaat hyötyisivät ohjelmointikielien perusrakenteiden oppimisesta. Nämä tehtävät vaativat oppilailta runsaasti mukautuvaa päättelyä ja strategista kompetenssia ja ovat varsinaisen matemaattisen aiheen ulkopuolella, joten vaikeita tehtäviä ei tulisi suunnata kaikille oppilaille. Lisäksi ratkaisun koodista tulee rakenteeltaan väistämättä monimutkainen ja siinä hyödynnetään useita ohjelmoinnin perusrakenteita. Motivoituneet oppilaat saivat vaikeat tehtävät tehtyä ongelmitta. Sen sijaan osalle oppilaista nämä olivat liian haastavia tehtäviä. Liian haastavat tehtävät vähentävät oppilaiden motivaatiota ja siksi keskitasoisia tehtäviä olisi hyvä olla riittävästi, jotta kaikki oppilaan saisivat onnistumisen kokemuksia ja oivaltamisen iloa ratkaisemalla sopivan tasoisia tehtäviä.

Kaiken kaikkiaan oppilaat olivat kiinnostuneita tietokoneista ja tekivät mielellään tehtäviä tietokoneella tai tableteilla. Kuitenkin luokka, jossa kaikilla oppilailla oli yhtäläinen mahdollisuus käyttää tietokonetta opiskeluun, suhtautuivat ohjelmointiin positiivisemmin. Ohjelmoinnin opiskelun kannalta olisikin tärkeää, että oppilailla olisi tietokoneet käytössä tunneilla, joilla ohjelmointia opiskellaan.

11 Pohdintaa

Ennen tutkimusta tutkimukseen osallistuvien oppilaiden vanhemmilta pyydettiin kirjallinen lupa osallistua tutkimukseen. Kirjallista lupaa haettiin myös koulujen johdolta. Oppilailla oli myös mahdollisuus kieltäytyä itse osallistumasta tutkimukseen. Tutkimuksessa haastateltavat oppilaat saivat itse ilmoittautua haastateltaviksi. Ennen tutkimusta oppilaille luotiin tunnukset Scratch-ympäristöön. Tunnukset linkitettiin tutkijan Scratch-tiliin. Näin tunnusten luomiseen ja tehtävien tallentamiseen ei tarvinnut oppilaiden henkilötietoja. Tutkimuksen aikana täytetyt lomakkeet kerättiin myös anonyymina. Tutkimuksen jälkeen oppilaiden tunnukset poistettiin ja kaikki muu aineisto tuhottiin.

Tutkimustulosten luotettavuuteen voivat vaikuttaa monet seikat. Tutkimuksessa oli mukana pieni määrä oppilaita, sillä tutkimus toteutettiin kehittävän ja laadullisen tutkimuksen menetelmiä hyödyntäen. Siksi tutkimustulokset eivät ole yleistettävissä koskemaan suurempaa oppilasjoukkoa. Tutkimustuloksissa ei otettu huomioon oppilaiden osaamista ennen tutkimuksen suorittamista, joten tulokset oppilaiden ymmärryksen ja osaamisen kehittymisestä ovat suuntaa antavia.

Tutkimuksessa kävi ilmi, että oppilailla oli vaikeuksia ymmärtää nelikulmioiden hierarkiaa. Tulosten valossa ohjelmointi ei auttanut oppilaita syventämään heidän ymmärrystään hierarkiasta. Sen sijaan oppilaat oppivat huomamaan erilaisten nelikulmioiden eroavaisuuksia ja yhtenevyyksiä. Tutkimustulokset eivät kuitenkaan täysin vastaa kysymykseen, miksi oppilailla oli vaikeuksia ymmärtää hierarkiaa. Tulosten pohjalta voidaan pohtia, ovatko alakoululaiset vielä liian nuoria ymmärtämään hierarkian kaltaista abstraktia käsitettä. Mikäli alakoululaisten on vaikeaa ymmärtää hierarkian käsitettä, asiaan olisi hyvä palata myöhemmin esimerkiksi yläkoulun puolella, kun oppilaan ovat hieman vanhempia ja valmiimpia ymmärtämään abstrakteja käsitteitä.

Oppilaat arvioivat tutkimuksessa kielentämiseen liittyvät tehtävät epämiellyttävimpinä. Tutkimustulokset eivät kuitenkaan kerro vastausta siihen, miksi juuri kielentämiseen liittyvistä tehtävistä pidettiin vähiten. Syitä tähän tulokseen voivat olla esimerkiksi se, että oppilaat kokivat kielentämisen haastavana tai he eivät olleet tottuneet vastaaviin tehtäviin. Vastuksen saaminen edellyttäisi tarkempaa tutkimusta kielentämisestä. Oppilaiden olisi hyvä oppia kielentämään matematiikan ratkaisuja jo alakoulussa kehittyäkseen matemaattisessa ilmaisussa.

Tutkimustulokset herättävät myös kysymyksiä ohjelmoinnin opettamisesta peruskoulussa. Ohjelmointi on osa opetussuunnitelmaa ja ohjelmoinnin opetus on kytketty opetussuunnitelmassa läheisesti matematiikan opettamiseen. Tässä tutkimuksessa tutkittiin matematiikan opetusta ohjelmoinnin avulla, mutta tuloksissa kävi ilmi, että oppilaat tekivät mieluiten tehtäviä, joka kehittivät ohjelmointitaitoja ja algoritmista ajattelua matematiikan tehtävien sijaan. Graafiset ohjelmointikielet tarjoavan paljon erilaisia mahdollisuuksia rakentaa muun muassa pelejä ja animaatioita. Opetussuunnitelma kuitenkin keskittyy pitkälti ohjelmoinnin matemaattiseen puoleen. Tutkimustulosten valossa oppilaat olivat kiinnostuneita etenkin ohjelmoinnin visuaalises-

ta puolesta kuten hahmojen rakentamisesta. Kouluissa voitaisiin nykyistä enemmän tarjota oppilaille mahdollisuuksia oppia ohjelmointia heidän oman kiinnostuksensa pohjalta. Ohjelmointia ja algoritmista ajattelua voitaisiin opetella nykyistä enemmän esimerkiksi kuvaamataidon tai musiikin oppituntien yhteydessä luomalla taiteellisia kokonaisuuksia ohjelmointia apuna käyttäen. Kaiken kaikkiaan ohjelmoinnin ja algoritmisen ajattelun opetuksen kytkemistä muihin oppiaineisiin tai omaksi oppiaineekseen olisi hyvä tutkia enemmän.

Tuloksista kävin myös ilmi luokkien erilainen suhtautuminen ohjelmointiin ja sen hyödyllisyyteen. Tuloksia saattaa osaltaan selittää opettajien erilainen tietotaso ohjelmoinnista ja tietokoneiden saatavuus luokissa. Tämä herättää kysymyksen onko kaikissa kouluissa yhtäläinen mahdollisuus oppia ohjelmointitaitoja ja kuinka paljon opettajien asenne vaikuttaa ohjelmoinnin oppimiseen etenkin peruskouluissa. Ohjelmoinnin opettaminen on melko uusi asia peruskoulussa, joten monella opettajalla ei välttämättä ole vielä valmiuksia opettaa ohjelmointia. Ohjelmoinnin ja algoritmisen ajattelun opettaminen vaatii jonkin verran perehtymistä aiheeseen. Jatkotutkimuksena olisi mielenkiintoista selvittää, onko luokanopettajilla ja matematiikan opettajilla valmiudet ja riittävä taitotaso opettaa ohjelmointia.

Lisäksi tutkimus herätti kysymyksen siitä, että onko kaikilla peruskoulun oppilailla yhtäläinen mahdollisuus oppia käyttämään tietokoneita oppimiseen ja kehittää omia tietotekniikan taitojaan. Etenkin tutkimuksen toteutus herätti kysymyksen, että voidaanko perusopetuksessa korvata tietokone tabletilla ja miten se vaikuttaa oppilaiden algoritmiseen ajatteluun ja ohjelmointitaitojen kehittymiseen. Tabletin haasteena on se, että tabletista puuttuu paljon ohjelmointia helpottavia ominaisuuksia. Esimerkiksi kirjoittaminen tabletilla on selvästi hitaampaa ja vaikeampaa verrattuna tietokoneeseen. Lisäksi esimerkiksi kopiointi on tietokoneen hiirellä huomattavasti nopeampaa verrattuna vastaavaan toimintoon kosketusnäytöllä. Kaikilla peruskoululaisilla olisikin hyvä olla mahdollisuus käyttää koulussa tietokonetta opiskellessaan ohjelmointia.

Lähteet

[1] Aho, A. V. (2011). Ubiquity symposium: Computation and Computational Thinking. Ubiquity, 2011(January), 8.

[2] Alaste, T. *Lineaarialgebra*. http://cc.oulu.fi/ tma/LAPIENI2016.pdf, Oulun yliopisto, 2016

[3] Berry, J. (suom. Sahlberg, P.) *Mitä matemaattinen ongelmanratkaisu on?*. Matematiikka - taitoa ajatella (toim. Seppälä, R.). Jyväskylä, Gummerus, 1994

[4] Chandler, D., Munday, R. programming. In A Dictionary of Media and Communication. : Oxford University Press. Retrieved 18 Sep. 2019, https://www-oxfordreference-com.libproxy.tuni.fi/view/10.1093/acref/9780191800986.001.0001/acref-9780191800986-e-3355.

[5] Dede, C. *If design-based research is the answer, what is the question? A commentary on Collins, Joseph, and Soloway in the JLS special issue on design-based research.*. The Journal of the Learning Sciences, 13(1), 105-114., 2004

[6] Dodero, J. M., Mota, J. M., and Ruiz-, I. *Bringing computational thinking to teachers' training: a workshop review*. In Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality. ACM, New York, NY, USA, Article 4, 2017

[7] Dreyfus, T. & Eisenberg, T. *On different facets of mathematical thinking*. The nature of mathematical thinking (toim. Sternberg, A & Ben.Zeev, T.). Mahwah (NJ): Erlbaum, 1994

[8] Eukleides, suom. Kahanpää, L., Aschan, P. *Alkeet*. Jyväskylä, 2011

[9] Fenn, R. *Geometry*. 2001

[10] Foerster, K. *Integrating Programming into the Mathematics Curriculum: Combining Scratch and Geometry in Grades 6 and 7*. In Proceedings of the 17th Annual Conference on Information Technology Education (SIGITE '16). ACM, New York, NY, USA, 91-96.,2016

[11] Ford, J. L. *Scratch programming for teens*. Course Technology PTR, 2008

[12] Hakkarainen, K., Lonka, K., Lipponen, L. *Tutkiva oppiminen, Älykkään toiminnan rajat ja niiden ylittäminen*. WSOY, 1999

[13] Joki, J. *Ulkoluvusta hahmottavaan geometriaan – Aineksia geometrian opetukseen erityisesti peruskoulussa*. Joensuun yliopisto matematiikan laitos, 2002

[14] Joutsen, E. *Eukleideen geometriaa*. https://jyx.jyu.fi/bitstream/handle/123456789/57340/URN:NBN:fi:jyu-201803161748.pdf?sequence=1, Jyväaskylän yliopisto, Matematiikan ja tilastotieteen laitos, 2018

[15] Joutsenlahti, J. *Lukiolaisen tehtäväorientoituneen matemaattisen ajattelun piirteitä* . Tampereen yliopisto, 2005

[16] Joutsenlahti, J., Kulju, P. *Multimodal Languaging as a Pedagogical Model—A Case Study of the Concept of Division in School Mathematics*. University of Tampere, 2017

[17] Kananen J. *Kehittämistutkimusopinnäytetyönä – Kehittämistutkimuksen kirjoittamisen käytännön opas* . Jyväskylän yliopisto, 2012

[18] Lepmann, L., Lepmann, T. *Kiehtovaa matematiikkaa*. 1997

[19] Malmi, L. *Ohjelmointiparadigmat*. Turun yliopisto, 2006, http://www.cs.hut.fi/Opinnot/T-106.3100/K2006/Luennot/Luento-1.pdf

[20] Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., Settle, A. (2014). Computational Thinking in K-9 Education. Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference - ITiCSE-WGR '14, 29.

[21] Opetushallitus, *Perusopetuksen opetussuunnitelman perusteet 2014*. https://www.oph.fi/sites/default/files/documents/perusopetuksen _opetussuunnitelman _perusteet _2014.pdf, 2014

[22] Pernaa, J. *Kehittämistutkimus opetusalalla*. PS kustannus Opetus 2000, 2013

[23] Polya, G. *How to solve it? A new aspect of mathematical method*. 2nd edition. Princeton (NJ): Princeton University Press, 1971

[24] Saez-Lopez, J., Roman-Gonzalez, M. and Vazquez-Cano, E. *Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools*. Elsevier, 2015

[25] Schoenfeld, A. *Learning to think mathematically: problem solving, matacognition and sense making in mathematics*. Handbook of research mathematics teaching and learning (toim. Grows, D. A. Mcmillan Publishing Co C, London , 1992

[26] *Scratch*. https://scratch.mit.edu/research

[27] Silfverberg, H. *Peruskoulun yläasteen oppilaan geometrinen käsitetieto*. https://trepo.tuni.fi/bitstream/handle/10024/66665/951-44-4718-2.pdf?sequence=1, Tampereen yliopisto, 1999

[28] Silfverberg, H., Portaankorva-Koivisto, P., Yrjänäinen, S. *Matematiikka kielenä ja kielikasvatuksena*. Teoksessa L. Jalonen, T. Keranto, & K.Kaila (Toim.),*Matematiikan ja luonnontieteiden opetuksen tutkimuksen tutkimuspäivät Oulussa 25.-26.11.2004. Matemaattisten aineiden opettajan taitotieto - haaste vai mahdollisuus?* Oulun yliopisto, Oulu: Kasvatustieteiden ja opettajankoulutuksen yksikkö, Oulun yliopisto

[29] Taisbak, C. M., van der Waerden, B. L. *Euclid*. https://www.britannica.com/biography/Euclid-Greek-mathematician, 2019

[30] Tuomi, J., Sarajärvi, A., *Laadullinen tutkimus ja sisällönanalyysi*. Tammi, 2018.

[31] Usiskin, Z., Willmore, E., Witonsky, D., & Griffin, J. *The Classification of Quadrilaterals: A Study of Definition*. Charlotte, NC: Information Age Publishing, 2008

[32] Väisälä, K. *Geometria*. Werner Söderström osakeyhtiö, Porvoo, 1959

[33] Weintrop, D., Wilensky, U. *Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms*. ACM Trans. Comput. Educ. 18, 1, Article 3, 2017

[34] Kuva helikulmioiden hierarkiasta, https://commons.wikimedia.org/wiki/File:Quadrilateral_hierarchy.png

[35] Weindrop, D., Wilensky, U. *Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms*. ACM Trans. Comput. Educ. 18, 1, Article 3, 2017

12 Liitteet

12.1 Liite 1

Anna tehtäville kouluarvosana (4-10). Anna arvosana 10 tehtävälle, josta pidit eniten, 9 tehtävälle, josta pidit seuraavaksi eniten ja niin edelleen.

☐	Kopioi neliön skripti ja vaihda kopion ensimmäiselle riville ”kun painetaan nuoli ylös”
☐	Saatko muutettua kopioitua skriptiä niin, että se piirtää neliön sijaan suorakulmion, joka ei ole neliö
☐	Mitä muutoksia teit, että sait piirrettyä suorakulmion
☐	Muuta kynän väriä. Voit päättää värin itse.
☐	Kopio neliön skripti. Saatko muutettua sitä niin, että se piirtää neliön sijaan neljäkkään, joka ei ole neliö
☐	Mitä muutoksia teit, että sait piirrettyä neljäkkään?
☐	Miten koodirivejä voi vähentää käyttämällä ohjaus-skriptejä?
☐	Kopio suorakulmion skripti. Saatko muutettua sitä niin, että se piirtää suunnikkaan, joka ei ole suorakulmio. Mitä muutoksia teit, että sait piirrettyä suunnikkaan?
☐	Muuta pingviini-hahmoa ja taustaa haluamallasi tavalla
☐	Lisää hahmollesi puhekupla, jossa hahmo kertoo, millaisen kuvion se on piirtänyt. Kaikkiin skripteihin täytyy lisätä puhekupla erikseen

Vastaa seuraaviin väittämiin. Ympyröi arvosana, joka kuvaa omaa mielipidettäsi parhaiten.

Mielestäni tehtävät olivat…	helppoja						vaikeita
	10	9	8	7	6	5	4
Mielestäni tehtävät olivat…	mukavia						ahdistavia
	10	9	8	7	6	5	4
Mielestäni tehtävät olivat…	mielenkiintoisia						tylsiä
	10	9	8	7	6	5	4
Ohjelmointitaidoista on minulle hyötyä tulevaisuudessa…	erittäin paljon						ei lainkaan
	10	9	8	7	6	5	4

Mikä mielestäsi oli parasta oppitunneilla?

Mitä olisi voinut tehdä paremmin?

12.2 Liite 2

Skripti = lyhyt tietokoneohjelma

Luokkani-osiosta löytyy skripti, jolla voi piirtää neliön. Kopioi se ja käytä sitä pohjana tehtäviä tehdessä.

Ohjeet skriptin kopiointiin:

Luokkani → Class Studio → Neliö → Katso sisälle → Remix → Vaihda projektin nimi → Tiedosto → tallenna nyt → Jaa

Jakamisen jälkeen projektin voi lisätä muiden luokkalaisten nähtäväksi (Luokkani → Class Studios → Lisää projekteja)

1. Kopioi neliön skripti ja muuta tapahtumaskriptiä niin, että ohjelma piirtää neliön, kun painetaan nuoli ylös
2. Saatko muutettua kopioitua skriptiä niin, että se piirtää neliön sijaan **suorakulmion**, joka ei ole neliö?
3. Mitä muutoksia teit, että sait piirrettyä suorakulmion?

4. Muuta väriä, jolla pingviini piirtää kuvioita
5. Kopioi neliön skripti. Saatko muutettua sitä niin, että se piirtää neliön sijaan **neljäkkään**, joka ei ole neliö?
6. Mitä muutoksia teit, että sait piirrettyä neljäkkään?

7. Muuta pingviini-hahmoa ja taustaa haluamallasi tavalla
8. Kopio suorakulmion skripti. Saatko muutettua sitä niin, että se piirtää **suunnikkaan**, joka ei ole suorakulmio. Mitä muutoksia teit, että sait piirrettyä neljäkkään?

9. Miten koodirivejä voi vähentää käyttämällä ohjausskriptejä?

10. Saatko hahmosi kertomaan, millaista kuviota se on piirtämässä? Hahmo voi kertoa tiedon puhekuplassa ja sanoa sen ääneen
11. Kopioi suunnikkaan skripti. Saatko muutettua sitä niin, että se piirtää suunnikkaan sijaan **puolisuunnikkaan**?
12. Miten sait skriptin piirtämään puolisuunnikkaan?

13. Kopio puolisuunnikkaan skripti. Saatko muutettua skriptiä niin, että se piirtää **nelikulmion**, joka ei ole, neliö, neljäkäs, suunnikas tai puolisuunnikas
14. Miten sait piirrettyä nelikulmion?

15. Tee ohjelmaan toinen hahmo
16. Muuta neliön skriptiä niin, että neliön sivun pituus on sattumanvarainen luku väliltä 20-100
17. Muuta neljäkkään skriptiä niin, että sivun pituuden ja kulmien suuruudet ovat sattumanvaraisia.
18. Lisää toiminto, jolla hahmo hyppää piirtämisen jälkeen satunnaiseen kohtaan ruudulla
19. Muuta suunnikkaan skriptiä niin, että sivun pituudet ja kulmien suuruudet ovat sattumanvaraisia.
20. Lisää Tieto-skriptillä muuttujat sivun pituuksille ja kulmien suuruuksille.
21. Tieto-skriptillä laatikko, joka kertoo sivun pituudet ja kulman suuruudet.

12.3 Liite 3