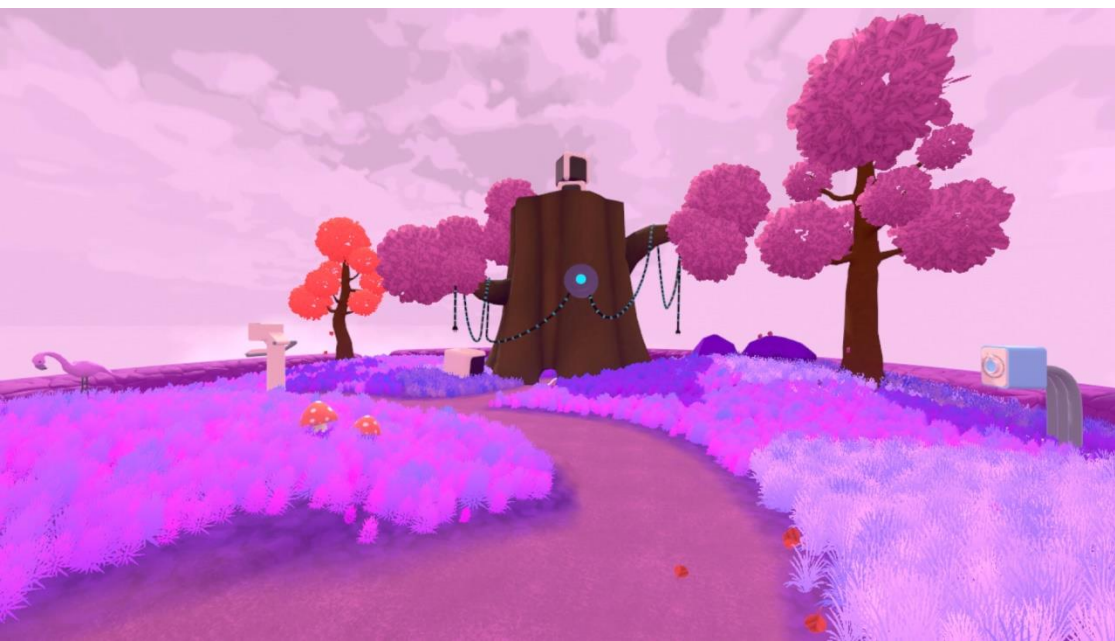


Markku Hartikainen

Ohjelmoinnin opetuspelin testaus ja kehitys

Tradenomi
Tietojenkäsittely
Syksy 2021



Tiivistelmä

Tekijä: Hartikainen Markku

Työn nimi: Ohjelmoinnin opetuspelin testaus ja kehitys

Tutkintonimike: Tradenomi

Asiasanat: ohjelmointi, opetuspelit, pelitestaus, mielikuvat, asenteet

Tässä opinnäytetyössä paneudutaan ohjelmoinnin opetuspelien suunnitteluun, kehitykseen sekä käyttöön opetuksessa. Erityisen mielenkiinnon kohteena on Kajaanin ammattikorkeakoulussa opiskelijaprojektina kehitetty DreamScript -ohjelmoinnin opetuspelejä ja sen toimivuus oppimisympäristönä. DreamScriptiä testattiin kahdessa kajaanilaisessa yläkoulussa ja pelitestaukseen osallistui yhteensä 71 oppilasta. Pelitestauksen ohessa tehtiin kaksivaiheinen kyselytutkimus, jonka avulla kerättiin tietoa DreamScriptin vaikutuksesta oppilaiden mielikuviin ohjelmoinnista.

Tulosten perusteella DreamScriptin pelaaminen sai oppilaat kokemaan ohjelmoinnin aiempaa helpommaksi, hauskemmaksi sekä itselleen sopivammaksi harrastukseksi. Peli myös kehitti oppilaiden kokonaisvaltaista mielikuvaa ohjelmoinnista myönteisemmäksi. DreamScript ei kuitenkaan pystynyt vähentämään ohjelmointiin negatiivisesti suhtautuneiden oppilaiden määrää. Tuloksista paljastui myös merkittäviä sukupuolten välisiä eroja ohjelmointiin suhtautumisessa. Pojat pitivät ohjelmointia helpompana, hauskempaan sekä itselle sopivampana ammattina kuin tytöt. Tytöistä 46 % ei pitänyt ohjelmointia lainkaan itselle sopivana ammattina, kun vastaava luku pojilla oli 10 %. On kuitenkin huomattava, että tyttöjen merkittävästi vähäisempi määrä vastaajissa voi vääristää tuloksia ja saada sukupuolierot näyttämään todellista suuremmilta.

Kyselytutkimuksen tulokset vahvistavat aiempaa tutkimusnäyttöä, jonka mukaan ohjelmoinnin opetuspelit ovat nuorille luonteva oppimisympäristö. Erityisen hyödyllisiä ohjelmoinnin opetuspelit ovat lähiopetukseen yhdistettynä sekä motivaation herättämiseen käytettynä. Ohjelmoinnin opetuspelit lisäävät oppilaiden aktiivisuutta ja harjaannuttavat muun muassa päättelykykyä, kommunikaatiotaitoja sekä kriittistä ajattelua. Ohjelmoinnin opetuspelien kehittäminen vaatii kuitenkin paljon pelitestausta sekä määrätietoista suunnittelua pelin tavoitteiden ja keinojen yhteensovittamiseksi.

Laadukas ohjelmoinnin opetuspelejä on tasapainoinen paketti hauskaa tekemistä sekä asiasisältöä. Se skaalautuu oppilaan tarpeiden mukaan tarjoten matalan kynnyksen aloittaa ja korkean katon kehittää omia sovelluksia. Laadukas ohjelmoinnin opetuspelejä sisältää myös sosiaalisia elementtejä sekä antaa mahdollisuuden reflektoida ja vertailla erilaisia ratkaisuja. Ohjelmoinnin opetuspeleissä voidaan hyödyntää myös fyysisiä elementtejä, kuten palikoita tai puheohjausta, jotka muuntavat ohjelmoinnin abstraktit käsitteet konkreettisemmiksi. Opettajan näkökulmasta korkeatasoinen ohjelmoinnin opetuspelejä sisältää paljon käyttömahdollisuuksia ja sen käyttöönottokustannukset ovat vähäiset.

Abstract

Author: Hartikainen Markku

Title of the Publication: Testing and Development of an Educational Programming Game

Degree Title: Bachelor of Business Administration

Keywords: programming, educational games, playtesting, conceptions, attitudes

This thesis focuses on the development and use of educational programming games in teaching. Of particular interest is the DreamScript programming game developed as a student project at Kajaani University of Applied Sciences and the game's functionality as a learning environment. DreamScript was playtested in Kajaani secondary schools where a total of 71 students attended in game testing. Playtesting sessions included a two-step survey which gathered information about the students' views of programming before and after playing the DreamScript.

Based on the survey results, after playing DreamScript, students perceived programming as easier, more fun and more suitable hobby. The game also changed students' general view of programming to more positive. However, DreamScript was unable to reduce the number of students who had a negative attitude towards programming. The survey results also revealed noteworthy gender differences in attitudes towards programming. Boys found programming easier, more fun and more suitable profession for themselves than girls. Overall, 46 % of girls did not consider programming to be a suitable profession for them at all, while the corresponding figure for boys was 10 %. However, a significantly smaller number of girls in the respondents may distort the results and make the gender differences look bigger than they really are.

The survey results were in line with the previous research evidence that the educational programming games are a natural learning environment for digital native youth. Programming games are especially useful when combined with contact teaching and when used as a motivational tool for beginning of the course. Educational programming games increase the students' involvement and they also improve students' reasoning, communication skills and critical thinking. However, development of the educational programming games is not straightforward and it requires a lot of playtesting and planning to achieve the intended goals of the game.

A high quality educational programming game is a balanced package of fun and instructive content. It adjusts according to the needs of the student, offering low floor to start and a high ceiling to create functions. A high standard educational programming game also includes social elements and provides an opportunity to reflect on and compare different solutions. The game can also utilize physical elements such as blocks or speech recognition that make the abstract programming concepts more concrete. From the teacher's perspective, a high quality educational programming game offers many use cases and its implementation costs are low.

Sisällys

1	Johdanto	1
2	Ohjelmoinnin opetuspelit oppimisympäristönä.....	2
2.1	Ohjelmoinnin opetuspelien hyödyt.....	2
2.2	Ohjelmoinnin opetuspelien suunnittelu ja kehittäminen	4
2.3	Ohjelmoinnin opetuspelien kritiikki	7
3	DreamScript -ohjelmoinnin opetuspelejä	9
3.1	DreamScript-pelin esittely.....	9
3.2	DreamScript-opetuspelelele testaus ja tutkimus.....	13
4	Tulokset	16
4.1	Taustamuuttujat.....	16
4.2	Mielikuvat ohjelmoinnista ennen pelitestausta.....	18
4.3	Oppilaan taustan vaikutus näkemyksiin ohjelmoinnista.....	20
4.4	Mielipiteet DreamScript-opetuspelelele	23
4.5	Oppilaan taustan vaikutus pelikokemukseen	26
4.6	Mielikuvat ohjelmoinnista pelitestausten jälkeen.....	29
4.7	Tutkimusasetelman kritiikki	33
5	DreamScriptin tulevaisuus.....	34
6	Yhteenveto	35
	Lähteet	36
	Liitteet	

1 Johdanto

Ohjelmointitaitojen merkitys kasvaa yhteiskunnassa, jossa palvelut automatisoituvat ja siirtyvät tietoverkkoihin. Harvalla on kuitenkaan kosketuspintaa siihen, miten arjessa käytetyt sovellukset todellisuudessa toimivat. Tietämättömyys sovellusten toiminnasta aiheuttaa ongelmia, kun sovellukset eivät toimikaan odotetulla tavalla. Ohjelmointitaitojen puute heikentää käyttäjien mahdollisuuksia löytää ratkaisuja tai keksiä uusia innovaatioita vanhojen sovellusten rinnalle. Ongelmatilanteiden välttämiseksi ihmisiltä olisi hyvä löytyä perustason ymmärrys ohjelmoinnista. Julkisessa keskustelussa ohjelmointi esitetään usein kansalaistaitona [1, s. 22].

Ohjelmoinnin opetuksella on yhä keskeisempi rooli suomalaisessa koulutusjärjestelmässä. Suomen opetus- ja kulttuuriministeriö käynnisti syksyllä 2020 Uudet lukutaidot - kehittämisohjelman, jonka tavoitteena on vahvistaa lasten ja nuorten medialukutaitoa, tieto- ja viestintäteknologista osaamista sekä ohjelmointiosaamista varhaiskasvatuksessa sekä esi- ja perusopetuksessa [2]. Viimeisimmässä, vuonna 2016 käyttöön otetussa perusopetuksen opetussuunnitelmassa ohjelmointitaidot kulkevat mukana kaikilla vuosiluokilla. Ohjelmoinnin maailmaan tutustuminen aloitetaan vaiheittaisten toimintaohjeiden laatimisella, josta siirrytään kohti graafisten ohjelmointiympäristöjen käyttöönottoa. Peruskoulun päättyessä tavoitteena on, että oppilas osaa hyödyntää luomiaan ohjelmia ongelmien ratkaisemisessa. Opetussuunnitelma on myös avoin uusille oppimisympäristöille ja tieto- ja viestintäteknologian hyödyntämiselle. Pelillisyyden katsotaan edistävän oppimisen iloa sekä vahvistavan edellytyksiä luovaan ajatteluun ja oivaltamiseen [3, s. 21].

DreamScript on Kajaanin ammattikorkeakoulussa opiskelijaprojektina kehitetty nykyaikainen ohjelmoinnin opetuspelejä. Tämän opinnäytetyön tavoite on selvittää DreamScriptin toimivuus oppimisympäristönä yli 12-vuotiaille lapsille. Hypoteesina on, että DreamScript avaa hauskaalla tavalla ohjelmoinnin käsitteistöä, vähentää ohjelmointiin liitettyjä negatiivisia mielikuvia sekä tekee ohjelmointiharrastuksen aloittamisesta aiempaa houkuttelevampaa. Tutkimuksen erityinen mielenkiinto kohdistuu siihen, millainen vaikutus DreamScriptin pelaamisella on oppilaiden mielikuviin ohjelmoinnista. Oppilaiden mielikuviin sisältyvät muun muassa käsitykset ohjelmoinnin vaikeudesta, hauskuudesta sekä itselle sopivuudesta. Ideaalitalanteessa DreamScript opettaa ohjelmoinnin peruskonsepteja, lisää pelaajien itsevarmuutta omiin ohjelmointikykyihin sekä kasvattaa kiinnostusta ohjelmoijan ammattia kohtaan. Tulosten ymmärtämiseksi tutkimuksessa selvitetään myös DreamScriptin onnistuneita sekä ongelmallisia pelimekaniikkoja sekä pelisuunnittelun ratkaisuja.

2 Ohjelmoinnin opetuspelit oppimisympäristönä

Hyötypelit pyrkivät vaikuttamaan pelaajan elämään myös pelin sisäisten tavoitteiden ulkopuolella [4]. Hyötypieleillä tavoitellaan esimerkiksi kiusaamisen vähentämistä, terveellisten ruokailutottumuksien edistämistä tai tapaturmien ehkäisyä [5] [6] [7]. Opetuspelit ovat hyötypelien alakategoria, sillä ne on tarkoitettu ensisijaisesti opetus- ja oppimiskäyttöön. Koska digitaalisten pelien keskeisiä ominaisuuksia ovat koukuttavuus, interaktio, kiehtovat grafiikat sekä hauskat aktiviteetit, opetuspelit ovat herättäneet runsaasti kiinnostusta tutkijoiden, pelikehittäjien, opettajien sekä oppilaiden keskuudessa. [8.] Pelipohjaisessa oppimisessä (Game Based Learning) oppimistavoitteita saavutetaan digitaalisten pelien avulla. Pelimaailmassa oppilaat aktiivisesti kokeilevat, tutkivat ja ratkovat ongelmia – parhaimmillaan taidot kehittyvät kuin varkein aktiviteettien ohessa. [9.]

Tässä luvussa tarkastellaan miten ohjelmoinnin opetuspelit toimivat, mitä hyötyä niiden käytöstä on, miten ohjelmoinnin opetuspelejä kehitetään sekä millaista kritiikkiä kyseiset pelit ovat saaneet.

2.1 Ohjelmoinnin opetuspelien hyödyt

Diginatiivilla sukupolvella on ennakkoluuloton suhtautuminen peleihin oppimisympäristönä. Nuoret käyttävät opetuspelejä mielellään ja kokevat myös hyötyvänsä niiden käytöstä [10, s. 337–338]. Opetuspelien on havaittu muun muassa laskevan opiskelun aloittamisen kynnystä, tekevän oppimisesta vähemmän aikaan ja paikaan sidottua sekä tuovan hauskoja ja kilpailullisia elementtejä oppimisprosessiin. Opetuspelit toimivat parhaimmillaan ideaalina oppimisympäristönä, koska interaktiivisuutensa vuoksi ne pakottavat soveltamaan tietoa. Asioiden ulkomuistiin painaminen ei enää riitä, kun pelaaja asetetaan ratkaisemaan ongelmia virtuaalimaailmassa. Ongelmanratkonta puolestaan parantaa oppilaiden keskinäistä vuorovaikutusta sekä osallisuutta oppitunnin tapahtumissa. [11, s. 367–368.]

Ohjelmoinnin opetuspelien käytöllä on mahdollista ratkoa tieto- ja viestintätekniikan opetuksen yleisimpiä haasteita, joita ovat teoreettisen tiedon siirtäminen käytännön harjoitteiksi, opetuksen pitäminen tarpeeksi mielenkiintoisena sekä abstraktien käsitteiden konkretisoiminen [12, s. 7]. Ohjelmoinnin opetuspelit edistävät käyttäjiensä ohjelmointitaitoja usealla eri tasolla. Ne paitsi opettavat ohjelmoinnin peruskonsepteja ja niiden hyödyntämistä ongelmanratkonnassa,

myös saavat ohjelmoinnin näyttäytymään positiivisemmassa valossa, kasvattavat kiinnostusta ohjelmointia kohtaan sekä lisäävät pelaajien luottamusta omaan kehittymiseensä. [13.] Tietotekniikan opetuksen kannalta ohjelmoinnin opetuspelien hyödyntämistä voidaan suositella etenkin motivaation herättämiseen ohjelmointikurssin alussa [12]. Ohjelmoinnin opetuspelien hyödyntäminen voi tarvittaessa olla hyvinkin vapaamuotoista. Kouluttajat voivat esimerkiksi antaa kurssikohtaisen suosituksensa siitä, mitä ohjelmoinnin opetuspelejä oppilaiden kannattaa pelata vapaa-ajallaan [14].

Ohjelmoinnin opetuspeleillä voidaan tukea jo pikkulasten tutustumista ohjelmointiin. Lapsille suunnatut ohjelmoinnin opetuspelit kehittävät laskennallista ajattelua sekä myönteisiä käyttäytymismalleja [15]. Rose ym. (2020) pystyivät parantamaan alakouluikäisten lasten ohjelmointikäytäntöjä kehittämänsä ohjelmoinnin opetuspelin avulla. Opetuspelin pelaamisen myötä lasten koodissa esiintyi vähemmän toistoa, monimutkaisia osioita tai saavuttamatonta koodia. [16.] Hyvä pelisuunnittelu yhdistettynä sopiviin oppimismenetelmiin mahdollistaa haastavien sisältöjen omaksumisen nuorella iällä [17]. Giannakoulus ym. (2018) tutkivat lapsille suunnatun ohjelmoinnin opetuspelin vaikutuksia viidesluokkalaisten ohjelmointitaitoihin. Tuloksena havaittiin, että opetuspeleillä auttoi oppilaita paitsi ymmärtämään ohjelmoinnin peruskonsepteja myös osallistumaan aktiivisemmin tunnin kulkuun. [18.] Ohjelmoinnin opetuspelit tekevät ohjelmoinnista tavoitteellista, hauskaa sekä entistä saavutettavampaa lapsille [19].

Ohjelmoinnin harjoittelu opettaa lukuisia käytännöllisiä taitoja, kuten kriittistä ajattelua, erilaisen suunnitelmien analysointia sekä ongelmanratkontaitoja [20, s. 87–98]. Ohjelmoinnin opetuspelit voivat tehdä ohjelmoinnin aloittamisesta nopeaa ja mielekästä. Esper ym. (2013) saivat 8–22-vuotiaista noviiseista koostuneen testiryhmän kirjoittamaan ja muokkaamaan Java-ohjelmointikieltä CodeSpells -ohjelmoinnin opetuspelissä vain 45 minuutissa. Tutkimuksen mukaan ohjelmoinnin opetuspelin pelaaminen sai testiryhmän kehittämään myönteisen suhtautumisen sekä omiin kykyihinsä että yleisesti ohjelmointihaasteisiin. [21.] Zhu ym. (2020) päätyivät vastaavaan tulokseen omassa tutkimuksessaan, jossa ohjelmoinnin opetuspelin pelaaminen kohensi merkittävästi opiskelijoiden itseluottamusta omia ohjelmointitaitoja kohtaan. Korkea itseluottamus on puolestaan yhteydessä tehokkaampiin ongelmanratkaisustrategioihin. [22.] Vihaivaisen ym. (2014) tutkimuskatsauksessa peliteemaiset opetusmenetelmät kasvattivat oppilaiden todennäköisyyttä läpäistä ohjelmoinnin alkeiskurssi lähes 11 prosenttia perinteiseen opettajaohjoitukseen opetusmenetelmään verrattuna [23].

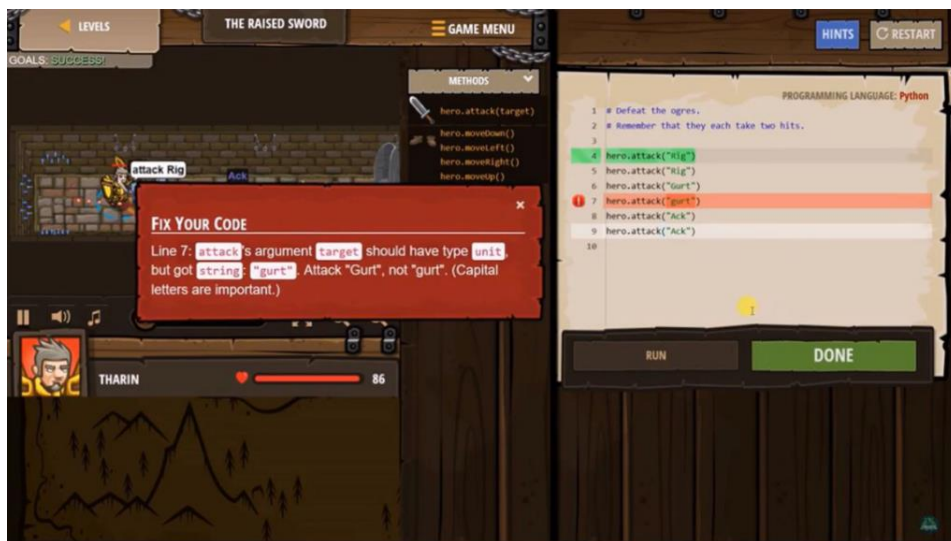
2.2 Ohjelmoinnin opetuspelien suunnittelu ja kehittäminen

Opetuspelien suunnittelun tueksi on luotu lukuisia yleispäteviksi tarkoitettuja malleja. Esimerkiksi Mitgutsch ja Alvarado loivat hyötypelejä koskevan suunnittelu- ja arviointikehyksen. Kehyksen mukaan opetuspelin laatuun vaikuttaa ennen kaikkea se, miten hyvin pelin osa-alueet tukevat pelin asettamaa opetuksellista tavoitetta. Tutkimusanalyyssissään Mitgutsch ja Alvarado erottivat kuusi erilaista opetuspelin laatua parantavaa elementtiä. Näitä elementtejä ovat opetuspelin tarkoitus, estetiikka, narratiivi, pelimekaniikka, kohderyhmä sekä informaationsisältö. Parhaimmillaan elementit tukevat toisiaan ja muodostavat yhtenäisen sekä mukaansa tempaivan pelikokemuksen, joka vaikuttaa pelaajaan ennakkoon valitulla tavalla. Ikävimmillään opetuspelin erilaiset elementit riitelevät keskenään ja pelin tarkoitus hukkuu pelaajan omien tarkoitusten sekä tulkintojen taakse. Opetuspelin elementtien tarkoituksenmukaisella suunnittelulla voidaan kuitenkin rajata mahdollisten pelikokemusten ja tulkintojen määrää. [4.]

Malliarakis ym. (2014) loivat opetuspelejä koskevien suunnittelumallien pohjalta oman mallinsa ohjelmoinnin opetuspelien suunnittelulle. Malliarakisen ym. suunnittelema malli korostaa tietoisia päätöksiä ohjelmoinnin opetuspelin tavoitteiden ja keinojen yhteensovittamiseksi. Mallin mukaisia päätöksiä ovat esimerkiksi oppimisstrategian sekä opetuksellisen sisällön valinta. Vasta kun nämä valinnat on suoritettu, voidaan ohjelmoinnin opetuspeleihin lähteä rakentamaan mielekkäitä aktiviteetteja. Pelkkä mielekkäiden sisältöjen suunnittelu ei kuitenkaan riitä. Käytännössä tarvitaan runsaasti pelitestausta ennen kuin voidaan riittävällä varmuudella sanoa, kuinka suuri osa opetuspelin pelaajista aidosti omaksuu pelin asettamat oppimistavoitteet. Oman haasteensa luovat erilaiset oppijat, sillä kun toisille sopii koodin itsenäinen kirjoittaminen, toiset mieluummin vetävät ja tiputtavat koodipaloja valikosta, kun taas osa vastaa mieluiten monivalinta-tehtäviin. Samoin opettajilla on opetuspelin suhteen omat toiveensa, sillä he yleensä haluavat muokata opetuspelin sisältöjä kunkin tunnin aihetta vastaavaksi. [24.]

Vaikka ohjelmoinnin opetuspelit vaihtelevat sisällöltään ja pelimekaniikoiltaan, on niissä nähtävillä paljon yhdistäviä elementtejä. Ohjelmoinnin opetuspeleissä on yleensä erilliset näkymät pelimaailmalle ja ohjelmakoodille. Käytännössä pelaaja siis joko kirjoittaa tai yhdistelee komponentteja ohjelmointi-ikkunassa, jonka jälkeen visuaalinen palaute annetaan pelinäkymässä. Joissain peleissä nämä näkymät ovat yhdistetty toisiinsa. Toinen ohjelmointipelejä yhdistävä vakioelementti on indikaattori, joka näyttää mitä kohtaa koodia ollaan tulkitsemassa. Tämä elementti paljastaa pelaajalle, millaisista rakenteista koodi koostuu ja millainen vaikutus kullakin elementillä on kokonaisuuden kannalta. Yleensä samalla indikaattorilla osoitetaan myös pelaajalle

jan tekemät virheet, jos koodi ei ole lainkaan tietokoneen tulkittavissa. [25, s. 6–7.] Kuvassa 1 on nähtävillä ohjelmoinnin opetuspeleille yhteisiä ja usein toistuvia elementtejä.



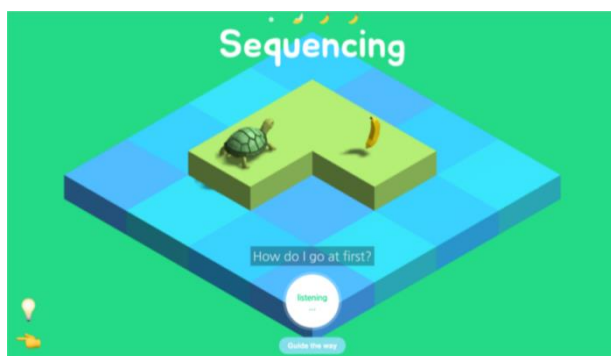
Kuva 1. Ruutukaappaus CodeCombat-pelistä, jossa esiintyy ohjelmoinnin opetuspeleille tyypillisiä elementtejä, kuten erillinen peli- ja ohjelmointinäköymä [25].

Ohjelmoinnin opetuspeleiden kehityksen keskeisimpiä haasteita on löytää tasapaino pelin viihdyttävien elementtien sekä opetuksellisen sisällön välillä [26, s. 80]. Tasapainon saavuttamisessa voidaan hyödyntää niin kutsuttua skaalautuvaa pelisuunnittelua. Skaalautuvan pelisuunnittelun avulla pyritään välttämään tilannetta, jossa opetuspelejä jää hetkellistä iloa tuottavaksi leluksi. Hyvin skaalautuva ohjelmoinnin opetuspelejä tarjoaa matalan kynnyksen aloittaa, mutta korkean katon luoda omia sovelluksia ja ratkaisuja. Kyse on kuin hiekkalaatikosta, jossa oppilas voi luoda itse mitä haluaa. Jos oppilaiden motivaatio toimia pelin parissa saadaan sisäsyntyiseksi, he voivat ylittää itsensä ja odotetun taitotasonsa. Lisäksi kun oppilas oppii ohjelmoimaan yhden toiminnon, sen voi siirtää sellaisenaan toiseen tilanteeseen tarjoten mahdollisuuden entistä monimutkaisemmille mekaniikoille. [27.] Parhaimmillaan avoimet ja skaalautuvat oppimisympäristöt mahdollistavat uusien opetuspeleiden luomisen ympäristön sisällä [28]. Avoimen kehittämissympäristön tarjoaminen ja sen käytön opettaminen vaativat kuitenkin paljon resursseja. Jo pelkkä ohjelmointikielen lukeminen ja kirjoittaminen ovat erillisiä kykyjä, jotka ohjelmoinnin opetuspeleiden täytyy tietoisesti opettaa erikseen [29, s. 61].

Oppimistulosten parantamiseksi opetuspeleiden tulisi saada pelaajansa kyseenalaistamaan ja reflektomaan oppimaansa. Reflektoinnilla tarkoitetaan oppijan kykyä analysoida ja mukauttaa omia toimintastrategioita tilanteen mukaan. Reflektointi lisää oppijan päätöksentekokykyä, so-

peutumista uusiin tilanteisiin sekä itsevarmuutta omaan tekemiseensä. Kyky reflektoida on erityisen tärkeää ohjelmoinnin kontekstissa, sillä ohjelmoija voi ratkaista saman ongelman lukemattomilla eri tavoilla. Ohjelmoinnin opetuspelit painottavat usein liiaksi vastausten oikeellisuutta erilaisten ratkaisujen löytämisen sijaan. Ohjelmoinnin opetuspeleissä tapahtuvan reflektion määrää voidaan lisätä esimerkiksi pelin aikana annetuilla vihjeillä tai sosiaalisilla elementeillä, joiden avulla pääsee näkemään ammattilaisten tai toisten opiskelijoiden tekemiä ratkaisuja. [25.]

Ohjelmoinnin opetuspelien keskeisimpänä etuna on niiden kyky mukautua kohderyhmänsä tarpeisiin. Esimerkiksi pikkulasten kynnystä toimia ohjelmoinnin opetuspelien parissa voidaan mataltaa ottamalla puheohjaus yhdeksi pelin vuorovaikutuksen välineeksi. Kuvassa 2 nähdään ruutukaappaus TurtleTalk-pelistä, jossa lapsi ohjaa puheellaan kilpikonnaa liikkeitä. Puheohjauksen on havaittu parantavan lasten keskittymistä opetuspeleihin ja siten helpottavan ohjelmoinnin konseptien sisäistämistä. [30.] Samoin virtuaalimaailman ulkopuoliset fyysiset elementit, kuten kuutiot, joilla ohjata pelimaailman tapahtumia, helpottavat abstraktien ohjelmoinnin konseptien käsittelemistä [13]. Opetuspelien tehokkuutta voidaan lisätä myös näennäisesti hyvin pienillä muutoksilla. Kao yms. (2021) tutkivat, miten ohjelmoinnin opetuspelissä seikkailevan sankarin ääni vaikuttaa oppimistuloksiin. Lopputuloksena havaittiin, että pelaajien suorituskyyky, motivaatio sekä immersio kasvoivat, jos sankarin ääni muistutti pelaajan omaa ääntä. Tällöin pelaaja pystyi samaistumaan paremmin opetuspelin päähenkilöön, mikä lisäsi pelaajan sitkeyttä ja omistautumista pelille. [31.]



Kuva 2. TurtleTalk-pelissä opetellaan ohjelmointia liikuttamalla kilpikonnaa puheohjauksella [30].

2.3 Ohjelmoinnin opetuspelien kritiikki

Ohjelmoinnin opetuspelejä kehitetään usein itsenäisesti, eikä tieto näiden käyttämistä opetusmenetelmistä ja niiden toimivuudesta yleensä kumuloidu [32]. Vaikka opetuspelit näyttävät lisäävän oppilaiden sisäsyntyistä motivaatiota opetettavaa asiaa kohtaan, kaikki tulkinnot eivät ole yhtä ruusuisia. Voi esimerkiksi olla, että opetuspelit lisäävät oppilaiden kiinnostusta enemmän opetuspelejä ja sen pelaamista kuin varsinaista opetettavaa asiaa kohtaan. Näkemys oppilaiden kiinnostuksen kohdistumisesta itse peliin on saanut tukea tutkimuksista, joissa ei ole löydetty korrelaatiota opetuspelien pelaamisen sekä sisäsyntyisen motivaation välillä. Chase ym. (2021) havaitsivat yläkouluikäisiä koskevassa tutkimuksessaan, että mitä enemmän ohjelmoinnin opetuspelejä sisälsi pelillisiä elementtejä, kuten ääniä, grafiikkaa ja juonta, sitä vähemmän oppilailla oli motivaatiota suoriutua vaikeista tehtävistä. Videopeleissä usein arvotetaan pelaajan suorituksia erilaisin pistein, tähdin ja kolikoin, mutta opetuspeleissä tämä voi kääntyä itseään vastaan. Tutkijoiden tulkinnan mukaan, mitä enemmän mittareita opetuspelejä antaa oppilaan suoritukselle, sitä raskaampaa oppilaan on säilyttää motivaatio epäonnistumisen jälkeen. Lisäksi pelilliset elementit kaappaavat oppilaan huomion ja lisäävät kognitiivista kuormaa, mikä saattaa häiritä oppimiseen keskittymistä. [33.]

Ohjelmoinnin opetuspelien tehokkuuden lisäksi niiden soveltuvuus itsenäiseen opiskeluun on kyseenalaistettu. Opetuspelien hyödyllisyyden on havaittu kasvavan lähiopetukseen yhdistettynä [26, s. 80]. Vailla opettajajohtoisia menetelmiä ohjelmoinnin opetuspelien on vaikea ilmaista oppilaille, mitä ruudulla tapahtuu ja miksi. Tämä lisää riskiä siihen, että oppilaat ymmärtävät väärin ja tekevät virheellisiä tulkintoja koodin toimintaperiaatteista. Pahimmillaan oppilaat saattavat kehittää jopa haitallisia oppimisstrategioita, joissa ei uskalleta kokeilla uusia ratkaisuja tai kyseenalaistaa aiemmin esitettyjä mallivastauksia. Ohjelmoinnin opetuspelit voivat kompensoida opettajajohtoisesta opastuksesta puutetta lisäämällä peliin paljon ohjeita, mutta toisaalta liika ohjeistus vie pelistä oivaltamisen ilon. [29.] Käytännössä opetuspelit joutuvat tekemään paljon kompromisseja, että mahdollisimman moni voisi käyttää peliä itsenäisesti. Varsinkin videopeleihin tottumaton käyttäjäkunta tarvitsee opetuspelejä käyttäessään erityistä huomiota [4, s. 126]. Kriittisten arvioiden mukaan ohjelmoinnin opetuspelien hyödyllisyyttä käsittelevissä tutkimuksissa ei huomioida riittävästi oppilaiden taustaa, persoonallisuuden piirteitä tai aiempaa osaamista [34].

Myös ohjelmoinnin opetuspelien markkinointiin liittyy lukuisia ongelmia. Koulutuksen järjestäjän on usein vaikea vertailla erilaisten ohjelmoinnin opetuspelien laatua tai mielekkyyttä. Suurin

osa ohjelmoinnin opetuspeleistä näyttää kattavilta ulospäin, mutta tarjoaa lopulta melko rajallisesti opetuksellista sisältöä [20, 87–98]. Vaikka opetuspelit olisi laadukkaasti toteutettu, se ei vielä sinänsä takaa pelin sopivuutta opetus- tai koulutuskäyttöön. Ohjelmoinnin opetuspelit ovat yleensä kehitetty kattamaan tietyt aihealueet, eikä niinkään kattamaan tietyt opetussuunnitelmat [14]. Myös oppitunnit asettavat tiukat aikarajat, joita ohjelmoinnin opetuspelien voi olla vaikea noudattaa. On keskeistä miettiä, ehtiikö opetuspelin esitellä, pelata ja ehtiikö siitä keskustella 45-minuutin tai 90-minuutin mittaisen oppitunnin aikana. Opetuspelin opetteluun ja käyttöönottoon on sitä korkeampi kynnys, mitä vähemmän käyttömahdollisuuksia tuote tarjoaa. Ideaalitulanteessa opetuspelit vaatii vain vähän ennakkovalmistelua, mutta tarjoaa paljon hyödyllistä sisältöä. Ennakkovalmistelun vähentämisessä keskeistä on, että opetuspelin laitteistovaatimukset ovat minimaalisia. Opetuspelit hylätään herkästi ensimmäisten teknisten ongelmien tai muiden vaikeuksien ilmaantuessa. [35, s. 20–21.]

Ohjelmoinnin opetuspelien saamaan kritiikkiin on vastattava entistä tarkoituksenmukaisemmallalla pelisuunnittelulla. Ratkaisuksi oppilaiden motivaatio- ja itsetunto-ongelmiin Chase ym. ehdottavat, että opetuspelit palkitsisivat pelaajia onnistumisten sijaan rakentavista ongelmanratkaisutaidoista, kuten tiedon etsinnästä, virheiden analysoinnista sekä uudelleen yrittämisestä [33]. Ohjelmoinnin opetuspelien tulisi painottaa, etteivät tietokoneet ole kaikkietäviä ja ylitse pääsemättömän monimutkaisia laitteita, vaan oikeasti tyhmiä apuvälineitä, jotka toimivat tismalleen niille annettujen ohjeiden mukaan. Tämän ajatusmallin omaksumisessa voi auttaa opetuspelit, joissa korjataan olemassa olevia, mutta virheellisiä sovelluksia. [29, s. 57.] Oppilaiden itsenäistä työskentelyä puolestaan auttavat tilannesidonnaiset ohjeet, jotka muuntuvat pelaajan kohtaamien vastoinkäymisten mukaan. Tilannesidonnaisten ohjeiden lisäksi ohjelmoinnin opetuspelien tulisi olla mahdollisimman kevyitä laitteistovaatimuksiltaan sekä mahdollisimman joustavia sisällöltään, että niitä voidaan käyttää tilanteen ja tarpeen mukaan. Etenkin komponenteista koostuvat ja avoimeen lähdekoodiin perustuvat ohjelmoinnin opetuspelit tarjoavat mahdollisuuden pelin soveltamiseen useissa eri yhteyksissä [36].

3 DreamScript -ohjelmoinnin opetuspeleli

Tässä luvussa tutustutaan DreamScript -ohjelmoinnin opetuspeleli taustaan, kehitykseen sekä pelitestaukseen. Luvussa esitellään myös tutkimusasetelma, jolla selvitettiin DreamScriptin toimivuus opetuspelelinä sekä pelin vaikutus oppilaiden mielikuviin ohjelmoinnista.

3.1 DreamScript-pelin esittely

Kajaanin ammattikorkeakoulussa opiskelijaprojektina kehitetty DreamScript tarjoaa nykyaikaisen lähestymistavan ohjelmoinnin opetuspeleihin. DreamScript on kokonaan ensimmäisestä persoonasta kuvattu tarinavetoinen seikkailu, missä pelaaja astuu työajallaan unelmoivan ohjelmoijan saappaisiin. Pelissä tutkitaan värikästä unimaailmaa, josta löytyy sekä ihmisten rakentamia että luonnonmukaisia elementtejä. Uusia pulmia kohdatessaan pelaajan täytyy miettiä, millaisia sääntöjä hän haluaa pelimaailman noudattavan. Tämän jälkeen pelaaja ohjelmoi nämä säännöt pelin virtuaalisilla tietokonepäätteillä. Matkan varrella tutustutaan tietokoneiden toimintaan, kerätään kumiankkoja sekä täydennetään ohjelmointiopasta. DreamScriptin alusta on PC, sillä pelaaja tarvitsee näppäimistön ohjelmakoodin muokkaamiseen. Kuvassa 3 esitellään DreamScriptin pelimaailmaa ja graafista tyyliä.



Kuva 3. Pelaaja voi manipuloida DreamScript-pelin maailmaa muokkaamalla ohjelmakoodia virtuaalisilla tietokonepäätteillä.

DreamScript muistuttaa ulkokuoreltaan ja pelattavuudeltaan perinteistä ensimmäisen persoonan seikkailu- ja toimintapeliä. DreamScript on lainannut elementtejä tunnetuista kaupallisista hittipeleistä, kuten Valve Softwaren kehittämästä Portalista [37], Jonathan Blowin tuottamasta The Witnessistä [38] sekä Arvi Teikarin palkintoja kahmineesta pulmapeli Baba Is Yousta [39]. Yhdistelemällä niin pulma-, seikkailu- kuin opetuspelien piirteitä DreamScript pyrkii pitämään kohderyhmänsä laajana ja siten madaltamaan peliin tarttumisen kynnystä. DreamScriptissä on runsaasti kaupallisista peleistä tuttuja viihteellisiä ominaisuuksia, kuten koko pelin kattava tarina, sukkela dialogi sekä kerättäviä esineitä. Viihteelliset ominaisuudet tasapainottavat DreamScriptin opetuksellista puolta, joka rakentuu monipuolisista ohjelmointipulmista. DreamScriptin ohjelmointikielenä toimii C#-ohjelmointikielen syntaksia mahdollisimman tarkkaan mukaileva kuvitteellinen ohjelmointikieli ”DreamScript”, jota pelin kaikki laitteet tottelevat. Ratkaistakseen DreamScriptin pulmia pelaajan on opittava ajattelemaan ohjelmoijan tavoin ja muokattava jo kirjoitettua koodia.

DreamScriptin kehitys aloitettiin tammikuussa 2020. Tarkoituksena oli luoda peli, joka rakentuu yksinkertaisen mutta omaperäisen idean varaan – DreamScriptissä pelaaja pystyy pelaamaan ja ohjelmoimaan peliä samaan aikaan. Alun perin yhden projektikurssin mittaiseksi työksi tarkoitettu hanke paisui useamman vuoden mittaiseksi. Kehitysprosessin aikana peliin on ehtinyt jättämään oman jälkensä kaiken kaikkiaan jo 16 opiskelijaa. Vaikka tekijätiimin vaihtuvuus on ollut suurta, pelin keskeisin mekaniikka on pysynyt samanlaisena. Virtuaalisten tietokonepäätteiden kanssa touhuamisen sekä unimaailmassa seikkailemisen on katsottu tarjoavan pelaajille hyvät lähtökohdat ohjelmoinnin kokeilemiseen turvallisessa ja kannustavassa ilmapiirissä. Seikkailun edetessä pelaaja tutustuu muun muassa ohjelmoinnissa käytettäviin laskutoimituksiin, eri muutujatyyppeihin sekä ehtolauseisiin. Ohjelmointikonseptien lisäksi DreamScriptissä sivutaan matematiikasta sekä fysiikasta tuttuja aiheita, kuten prosentteja, massoja sekä kappaleen tilavuutta. Videopelit yleensäkin opettavat lukuisia taitoja, mitä niiden varsinaisessa suunnittelussa ei ehkä ole tarkoitettu tai otettu huomioon [40, s. 26–27].

DreamScriptin sisältö on jaettu 17 kenttään, joista kukin sisältää lukuisia erillisiä pulmia. Koko pelin läpipelaamiseen tarvittava aika on noviisilta ohjelmoijalta noin 3–5 tuntia. Taulukkoon 1 on koottu kussakin pelikentässä tarvittavat ohjelmointikonseptit, jotka pelaajan täytyy sisäistää edetäkseen pelissä. Taulukosta nähdään, että DreamScript esittelee pelaajalle uusia konsepteja säännöllisin väliajoin. Käytännössä jokaista uutta esitettyä ominaisuutta sovelletaan muutaman kentän ajan ennen siirtymistä seuraavaan konseptiin. Paloittain tapahtuva opetus paitsi pitää pelin mielenkiintoisena myös varmistaa sen, ettei pelaaja jää informaatiotulvan alle. DreamSc-

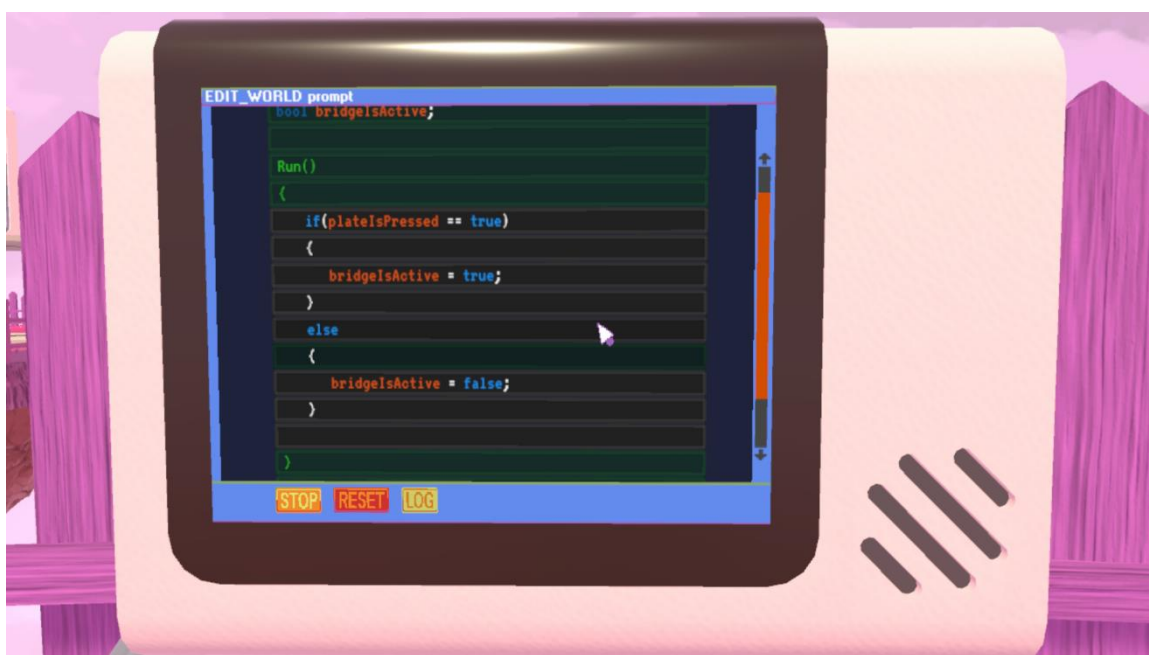
riptin konsepti ja pelimekaniikat tarjoaisivat mahdollisuuden myös paljon kattavampaan ohjelmointikonseptien käsittelyyn, mitä parin vuoden ajan kehitetyssä projektissa on käytännössä ehditty toteuttaa.

Opetuksellinen sisältö	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Boolean – muuttujat	X	X			X	X	X	X	X	X	X	X	X	X	X		X
Double – muuttujat		X	X	X	X	X		X		X	X	X		X	X	X	X
Prosenttilaskut	X	X	X	X	X	X											X
Ehtolauseet							X	X	X	X	X	X	X	X	X		X
Kommentit												X		X			X
Inkrementti ja dekrementti														X	X	X	X
Vertailuoperaattorit														X	X		X

Taulukko 1. DreamScript-pelin sisältämät kentät ja niiden opetuksellinen sisältö.

Pedagogisesti mielekkäät ja tarkoituksenmukaiset oppimisympäristöt vaativat tarkkaa suunnittelua. Jotta DreamScript toimisi opetuspelinä, sen kehityksessä on noudatettu kolmea keskeistä sääntöä. Ensimmäinen sääntö on, että pelaajaa kannustetaan aina kokeilemaan ja leikkimään koodin parissa. Koska saman ominaisuuden voi ohjelmoida lukuisilla eri tavoilla, lähes jokaiseen DreamScriptin pulmaan on useita ratkaisuja. Erilaisten ratkaisujen kokeilemiseen kannustetaan tekemällä pelaajan säätämistä muuttujista mahdollisimman luovia ja mielenkiintoisia. DreamScriptissä pelaaja pääsee muokkaamaan muun muassa pelimaailman painovoimaa, omaa nopeutta ja kokoaan, hissien vauhtia, lukkojen salasanoja sekä esineiden määrää, painoa, tilavuutta, paikkaa ja olomuotoa. Kokeiluun kannustaa myös pelin sisältämä dialogi, jossa tietokone kommentoi pelaajan tekemiä ratkaisuja joko humoristisesti tai kannustavaan sävyyn. Näillä eväillä pelaaja oppii toimimaan koodin parissa eikä pelkää myöskään pysähtyä miettimään ratkaisuja pitemmäksi aikaa.

Toinen keskeinen tekijä DreamScriptin suunnittelun kannalta on pelin vaikeustason optimointi. DreamScriptin haasteena on opettaa pelaajalle sekä itse pelin pelaamiseen liittyvät toiminnot, kuten liikkuminen ja vuorovaikutus pelimaailmassa, että tavoitteellinen opetuksellinen sisältö eli ohjelmoinnin perusteet. DreamScript pyrkii opettamaan pelaamista ja ohjelmointia yhtäaikaista sekä mahdollisimman immersiiivisesti eli pelaajan osallisuuden astetta lisäävästi. Immersioita edistävät pelin helppokäyttöisyys, tunteisiin vetoavuus sekä tarkkaavaisuuden vaatiminen useilla aisteilla. [41, s. 1297–1300.] DreamScriptissä vaikeusastetta pyritään lisäämään tasaisesti, mutta pelaajan kokeilunhalua kunnioittaen. Pelaaja voi esimerkiksi heittää tavaroita ensimmäisestä kentästä alkaen, mutta tätä ominaisuutta tarvitaan vasta neljännessä kentässä, jossa ominaisuus tietoisesti esitellään. Samoin pelaaja voi ohjelmoida virtuaalisiin tietokonepäätteisiin välittömästi ehtolauseita, mutta opetuksellisesti ehtolauseet tulevat vasta kuudennessa kentässä. Kuvasta 4 nähdään esimerkki siitä, miltä ehtolause näyttää DreamScriptin virtuaalisessa kehitysympäristössä.



Kuva 4. DreamScript-pelissä eteneminen vaatii muun muassa ehtolauseiden hallintaa.

Kolmas sääntö, jota seuraamalla DreamScriptistä on pyritty tekemään laadukas ohjelmoinnin opetuspelejä, koskee informaation saatavuutta. Opetuspeleissä on usein se ongelma, ettei koskaan voida olla varmoja siitä, onko pelaaja todella sisäistänyt kulloinkin esitetyn asian. Liian helpot ongelmat mahdollistavat sen, että pelaaja pääsee etenemään pelissä vain arvaamalla. Liian

vaikeat pulmat puolestaan mahdollistavat sen, että pelaaja turhautuu ja alkaa vierastaa käsitellyä aihetta. DreamScriptissä tätä ilmiötä pyritään lieventämään pitämällä pulmat ja pelimaailma leikkillisinä. Leikkisyys käy ilmi esimerkiksi kuvan 5 pelitilanteesta, jossa ohjelmakoodin sisältämä bugi esitetään pelaajalle kirjaimellisena ötökkänä. DreamScriptin perimmäinen tavoite onkin saada pelaajat tutustumaan kokeillen ohjelmoinnin maailmaan, eikä koodin täydellinen ymmärtäminen ole pelin tavoitteena tai vaatimuksena. Ymmärtämisen parantamiseksi DreamScriptin sisältämiin virtuaalisiin tietokonepäätteisiin on kuitenkin upotettu ohjelmointiopas, josta pelaaja löytää lisätietoa ohjelmoinnista sekä kulloinkin käsillä olevasta ongelmasta. Käytännössä DreamScriptin kentät ovat siis suhteellisen helppoja, mutta toisaalta pelaajalle tarjotaan aina myös ylimääräistä informaatiota, jolla syventää omaa osaamistaan.



Kuva 5. DreamScriptissä pelaaja joutuu moniin kiperiin tilanteisiin, kuten taistelemaan jättimäistä bugia vastaan.

3.2 DreamScript-opetuspelin testaus ja tutkimus

DreamScript-opetuspeleä testattiin kajaanilaisissa yläkouluissa viidellä eri luokalla syksyllä 2020. Pelitestauksen ohessa tehtiin kyselytutkimus, jolla kartoitettiin paitsi DreamScriptin toimivuutta opetuspelinä, myös pelin vaikutuksia oppilaiden mielikuviin ohjelmoinnista. Tutkimuksen kontekstissa mielikuvilla tarkoitetaan muun muassa oppilaiden käsityksiä ohjelmoinnin vaikeudesta,

hauskuudesta sekä itselle sopivuudesta. Tutkimushypoteesina oli, että viihdyttävänä ja interaktiivisena oppimisympäristönä DreamScript lisää pelaajien positiivisia mielikuvia ohjelmointia kohtaan. Hypoteesin mukaan yläkoululaiset kokisivat ohjelmoinnin hauskempana, helpompana sekä itselle sopivampana harrastuksena pelitestausten jälkeen kuin ennen pelitestausta.

Tutkimuksen perusjoukko valikoitui DreamScript-pelin kohderyhmän mukaan – eli lapsiin ja nuoriin, joille ei ole vielä kertynyt merkittävästi aiempaa ohjelmointikokemusta. DreamScriptin kohderyhmä ei sinänsä rajoitu lapsiin, mutta käytännön toteutuksen vuoksi tutkimuksen perusjoukko oli järkevä rajata alaikäisiin. Lopulta tutkimuksen otoskooksi saavutettiin 71 perusjoukkoon kuuluvaa yläkoululaista. Tutkimuksen taustamuuttujina huomioitiin pelitestaajien sukupuoli, harrastuneisuus videopeleissä sekä aiempi ohjelmointikokemus. Tarkoituksenmukaisesti valituilla taustamuuttujilla pyritään helpottamaan tulosten tulkintaa. Esimerkiksi oletettavaa on, että jos oppilaalla on hyvin vähän kokemusta videopeleistä, ei opetuspelistä todennäköisesti saada kaikkea hyötyä irti. Hyöty voi olla rajallinen myös niiden oppilaiden kohdalla, joille on jo kertynyt huomattavasti aiempaa ohjelmointikokemusta. Jos valittujen taustamuuttujien jakaumat otoksessa ovat likipitään samanlaiset kuin perusjoukossa, voidaan tätä pitää hyvänä merkinä tulosten yleistettävyyden kannalta [42, s. 30].

Tutkimuksen aineistonkeruumenetelmänä toimi kaksivaiheinen ja kontrolloitu kysely, jolla pelaajien käsityksiä ohjelmoinnista kartoitettiin ennen ja jälkeen pelisession. Kyselyiden etuna on niiden kyky tuottaa nopeasti laaja ja tilastoitavissa oleva tutkimusaineisto [43, s. 190]. Koska ohjelmoinnin opetus kuluu nykymuotoiseen peruskoulun opetussuunnitelmaan, oli tutkimukseen helppo saada osallistujia kajaanilaisista yläkouluista. Käytännössä pelitestaustilanteet järjestettiin viidessä eri sessiossa matematiikan sekä ATK:n oppitunneilla. Aluksi oppilaat vastasivat kyselyn ensimmäiseen osioon, jolla selvitettiin vastaajien taustamuuttujia sekä mielikuvia ohjelmoinnista. Seuraavaksi oppilaat saivat noin 50 minuuttia aikaa pelata DreamScriptiä ja muodostaa mielipiteensä pelistä. Lopuksi oppilaat vastasivat kyselyn jälkimmäiseen osioon, jolla kartoitettiin oppilaiden mielipiteitä pelistä sekä toistettiin samat oppilaiden ohjelmointimielikuviin liittyvät kysymykset. Näin DreamScriptin vaikutus oppilaiden mielikuviin ohjelmoinnista saatiin käännettyä mitattavaan muotoon. Kyselytutkimus on nähtävillä kokonaisuudessaan liitteestä 1.

Tutkimusasetelmassa oli keskeistä huomioida tutkittavien alaikäisyys. Alle 15-vuotiaiden oppilaiden kohdalla päätöksen tutkimukseen osallistumisesta tekevät ensisijaisesti oppilaan huoltajat. Kuitenkin suurille vastaajamäärille kohdennettavissa kyselytutkimuksissa huoltajien informointi ennakoon voidaan katsoa riittäväksi toimenpiteeksi. Toimenpide on riittävä myös tutkimuksissa, joissa ei käsitellä alaikäisten tutkittavien henkilötietoja. [44, s. 9–10.] Koska tässä

tutkimuksessa vastaajilta ei kerätty muita henkilötietoja kuin sukupuoli, nähtiin riittäväksi toimenpiteeksi informoida oppilaiden huoltajia ennakkoon mahdollisuudesta kieltäytyä tutkimuksesta. Tutkimusprosessissa noudatettiin myös muita yleisiä ihmistieteiden eettisiä periaatteita. Eettisiin periaatteisiin lukeutuvat muun muassa tutkittavien oikeus kieltäytyä osallistumasta, oikeus keskeyttää osallistuminen sekä oikeus saada tietoa tutkimuksen tavoitteista sekä henkilötietojen käsittelystä. Osallistujia informoitiin oikeuksistaan sekä suullisesti pelitestauksen alussa että kirjallisesti kyselytutkimuksen ohjeissa.

4 Tulokset

Tässä luvussa raportoidaan ja analysoidaan DreamScriptin pelitestauksen ohessa kerätyn kyselytutkimuksen tulokset. Luku alkaa perusjoukon esittelyllä, jossa käydään läpi muun muassa pelitestaajien sukupuolijakauma, peli- ja ohjelmointikokemus sekä ohjelmointimielikuvat. Seuraavaksi tarkastellaan, miten pelitestaajien tausta vaikuttaa heidän ohjelmointimielikuviinsa. Tämän jälkeen huomio kiinnitetään siihen, miten pelitestaajat arvioivat omaa pelikokemustaan DreamScriptin parissa ja miten pelitestaajien tausta vaikuttaa tähän arvioon. Lopuksi selvitetään, vaikuttiko DreamScriptin pelaaminen oppilaiden ohjelmointimielikuviin positiivisesti tai negatiivisesti. Luku pyrkii avaamaan aineiston mahdollisimman perusteellisesti sekä käsittelemään aineistosta nousevat keskeisimmät havainnot, riippuvuussuhteet ja näkökulmat. Luvun päätteeksi pohditaan myös saatujen tutkimustulosten luotettavuutta.

4.1 Taustamuuttujat

DreamScriptin pelitestaukseen osallistuneista 71 yläkoululaisesta 84 % oli poikia ja 16 % tyttöjä. Vahvasti poikiin painottunut otos selittyy todennäköisesti sillä, että valtaosa pelitestaussessioista järjestettiin ATK:n oppitunneilla, mikä on perinteisesti poikien suosima valinnaisaine. Taulukosta 2 nähdään, että vastaajista 90 % kertoi pelaavansa videopelejä vapaa-ajallaan vähintään muutaman kerran viikossa. Ainoastaan 3 % oppilaista ilmoitti, ettei pelaa videopelejä lainkaan. Tulos tukee näkemystä siitä, että videopelit ovat luonteva oppimis- ja toimintaympäristö lasten elämässä. Vastauksissa oli kuitenkin nähtävillä selkeää eroa sukupuolten välillä. Tytöistä 64 % ilmoitti pelaavansa videopelejä vähintään muutaman kerran viikossa, kun vastaava luku pojilla oli 95 %. Samoin tytöistä 18 % ilmoitti, ettei pelaa videopelejä lainkaan, kun taas pojista kukaan ei valinnut tätä vaihtoehtoa.

Sukupuoli	Pelaan videopelejä vapaa-ajallani					Yhteensä
	En lainkaan	Kerran kuukaudessa tai harvemmin	Muutaman kerran kuukaudessa	Muutaman kerran viikossa	Päivittäin	
Mies	0,0 %	1,7 %	3,4 %	16,9 %	78,0 %	100,0 %
Nainen	18,2 %	0,0 %	18,2 %	18,2 %	45,5 %	100,0 %
Tyhjä	0,0 %	0,0 %	0,0 %	0,0 %	100,0 %	100,0 %
Yhteensä	2,8 %	1,4 %	5,6 %	16,9 %	73,2 %	100,0 %

Taulukko 2. Videopelien pelaamisen säännöllisyys sukupuolen mukaan jaoteltuna.

Tulosten perusteella nuoret viettävät säännölliset aikaa videopelien parissa. Ohjelmoinnin opetuspelit eivät kuitenkaan kuulu oppilaiden suosikkigenreen, sillä vastaajista 61 % ilmoitti, ettei ole koskaan aiemmin pelannut ohjelmoinnin opetuspelejä. Taulukosta 3 nähdään, että alle viisi tuntia ohjelmoinnin opetuspelejä kertoi pelanneensa 32 % ja yli viisi tuntia 7 % vastaajista. Tulosten perusteella peruskoulun opetuksessa ei siis ainakaan säännönmukaisesti hyödynnetä ohjelmoinnin opetuspelejä. Tämä on yllättävää, sillä aiempien tutkimustulosten valossa opetuspelit ovat hyödyllinen ympäristö ohjelmoinnin opetteluun käynnistämiseen. Ohjelmoinnin opetuspelejä olisi käytännöllistä hyödyntää etenkin peruskouluissa, joissa opettajilla ei lähtökohtaisesti ole ohjelmointitaitausta. Tuloksia saattaa kuitenkin vääristää se, miten oppilaat määrittelevät ohjelmoinnin opetuspelejä. Jos tunnilla käytettävä sovellus ei erityisemmin sisällä graafisia tai pelillisiä elementtejä, oppilaat voivat tulkita ratkovansa enemmän tehtäviä kuin pelaavansa opetuspelejä.

Sukupuoli	Olen pelannut ohjelmoinnin opetuspelejä					Yhteensä
	En lainkaan	Alle tunnin	Yli tunnin	Yli viisi tuntia	Yli kymmenen tuntia	
Mies	57,6 %	15,3 %	18,6 %	5,1 %	3,4 %	100,0 %
Nainen	81,8 %	9,1 %	9,1 %	0,0 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	60,6 %	14,1 %	18,3 %	4,2 %	2,8 %	100,0 %

Taulukko 3. Ohjelmoinnin opetuspelien pelaamisen määrä sukupuolen mukaan jaoteltuna.

Taulukosta 4 nähdään oppilaiden ohjelmointikokemuksen määrä sukupuolen mukaan jaoteltuna. Koska ohjelmointitaidot kuuluvat nykykuotoisen peruskoulun opetussuunnitelmaan, ei ole yllättävää, että ohjelmointia ilmoitti vähintään kokeilleensa 76 % oppilaista. Näistä ohjelmointia kokeilleista oppilaista 18 % vastasi ohjelmoineensa yksinkertaisen sovelluksen ja 3 % monimutkaisen sovelluksen. Yksikään vastaajista ei kuitenkaan ilmoittanut harrastavansa ohjelmointia, mikä oli vastausvaihtoehtoista viimeisin. Sen sijaan 24 % oppilaista ei ollut kokeillut lainkaan ohjelmointia, mikä herättää kysymyksen, missä yhteydessä ohjelmoinnin kokeilu on muilla oppilailla tapahtunut. Todennäköisesti vastauksiin vaikuttavat oppilaiden erilaiset määrittelyt sille, mikä lasketaan ohjelmoinnin kokeilemiseksi. Tavassa määritellä saattaa olla myös eroa sukupuolten välillä, sillä tytöistä 55 % ilmoitti, ettei ollut kokeillut lainkaan ohjelmointia, kun vastaava luku pojilla oli 19 %.

Sukupuoli	Olen kirjoittanut ohjelmakoodia				Yhteensä
	En lainkaan	Olen kokeillut ohjelmointia	Olen ohjelmoinut yksinkertaisen sovelluksen	Olen ohjelmoinut monimutkaisen sovelluksen	
Mies	18,6 %	59,3 %	18,6 %	3,4 %	100,0 %
Nainen	54,5 %	27,3 %	18,2 %	0,0 %	100,0 %
Tyhjä	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	23,9 %	54,9 %	18,3 %	2,8 %	100,0 %

Taulukko 4. Ohjelmointikokemuksen määrä sukupuolen mukaan jaoteltuna.

4.2 Mielikuvat ohjelmoinnista ennen pelitestausta

Taustamuuttujien keräämisen jälkeen kyselytutkimuksessa kohdennettiin huomio oppilaiden mielikuviin ohjelmoinnista. Ensimmäisenä kysyttiin, miten sopivana ammatina tai harrastuksena oppilaat kokivat ohjelmoinnin. Taulukosta 5 nähdään, että 44 % vastaajista suhtautui neutraalisti ohjelmoinnin sopivuuteen itselle. Kokonaisuutena oppilailla oli kuitenkin melko negatiivinen näkemys ohjelmoinnin soveltuvuudesta omaksi harrastukseksi. Ainoastaan 14 % vastaajista katsoi ohjelmoinnin olevan itselleen edes jossain määrin sopiva ammatti tai harrastus. Sen sijaan epäsovivana tai suhteellisen epäsovivana harrastuksena ohjelmoinnin koki 42 % oppilaisista. Merkillepantavaa on myös sukupuolten välinen huomattava ero vastauksissa. Tytöistä yksikään ei kokenut ohjelmoinnin olevan itselle sopiva tai suhteellisen sopiva harrastus. Pojista ohjelmoinnin koki vähintään jossain määrin sopivaksi harrastukseksi 17 %. Samoin pojista ainoastaan 10 % ilmoitti, ettei ohjelmointi ole lainkaan itselle sopiva harrastus, kun vastaava luku tyttöillä oli jopa 46 %. Tulosten perusteella pojat suhtautuvat tyttöjä huomattavasti neutraalimmin ohjelmoinnin mahdollisuuteen tulevana harrastuksena tai ammatina.

Sukupuoli	Kuinka sopivana ammatina / harrastuksena pidät ohjelmointia itsellesi?					Yhteensä
	1. En lainkaan sopivana	2	3	4	5. Erittäin sopivana	
Mies	10,2 %	27,1 %	45,8 %	15,3 %	1,7 %	100,0 %
Nainen	45,5 %	27,3 %	27,3 %	0,0 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	15,5 %	26,8 %	43,7 %	12,7 %	1,4 %	100,0 %

Taulukko 5. Ohjelmointiammatin tai -harrastuksen koettu sopivuus itselle sukupuolen mukaan jaoteltuna.

Vaikka tyttöjen mielenkiinto ohjelmointia kohtaan oli selkeästi vähäisempi kuin poikien, ei se välttämättä tarkoita sitä, että sukupuolilla olisi erilainen mielikuva ohjelmoinnista. Tytöt ja pojat voivat yksinkertaisesti pitää eri asioista. Esimerkiksi kysyttäessä, miten vaikeana oppilaat koke-

vat ohjelmoinnin, vastaukset olivat sukupuolten välillä jo selkeästi yhdenmukaisempia. Taulukosta 6 nähdään, että sekä tytöistä että pojista miltei puolet piti ohjelmointia suhteellisen vaikeana. Vajaa kolmasosa sekä tytöistä että pojista suhtautui neutraalisti ohjelmoinnin vaikeuteen. Voidaan päätellä, ettei tyttöjen selkeästi vähäisempää mielenkiintoa ohjelmointia kohtaan voida selittää pelkästään ohjelmoinnin koetulla vaikeudella. Tytöt kyllä arvioivat poikia useammin ohjelmoinnin erittäin vaikeaksi, mutta ero on niin pieni, että se saattaa selittyä kokonaan tyttöjen huomattavasti vähäisemmällä määrällä vastaajissa. On otettava tarkasteluun muitakin ohjelmointiin liitettyjä käsityksiä.

Kuinka vaikeana pidät ohjelmointia?						
Sukupuoli	1. En lainkaan vaikeana	2	3	4	5. Erittäin vaikeana	Yhteensä
Mies	1,7 %	15,3 %	32,2 %	47,5 %	3,4 %	100,0 %
Nainen	0,0 %	9,1 %	27,3 %	45,5 %	18,2 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	1,4 %	14,1 %	32,4 %	46,5 %	5,6 %	100,0 %

Taulukko 6. Ohjelmoinnin koettu vaikeus sukupuolen mukaan jaoteltuna.

Seuraavaksi kyselytutkimuksessa selvitettiin yläkoululaisten näkemyksiä ohjelmoinnin hauskuudesta. Taulukosta 7 nähdään, että tytöistä ja pojista yhteensä 45 % suhtautui neutraalisti ohjelmoinnin hauskuuteen. Kuitenkin pojista ohjelmoinnin koki suhteellisen tai erittäin hauskana 46 %, kun vastaava luku tytöillä oli 15 %. Tytöistä puolestaan 30 % näki ohjelmoinnin epähauskana, kun pojista näin ajatteli vain 11 %. Taulukoiden 6 ja 7 vastauksista voidaan päätellä, että pojat kokevat ohjelmoinnin sekä vaikeaksi että hauskaksi. Tytöt puolestaan kokevat ohjelmoinnin vaikeaksi, mutta ei niinkään hauskaksi. Kaiken kaikkiaan oppilaiden suhtautuminen ohjelmoinnin hauskuuteen oli enemmän myönteistä kuin kielteistä.

Kuinka hauskana pidät ohjelmointia?						
Sukupuoli	1. En lainkaan hauskana	2	3	4	5. Erittäin hauskana	Yhteensä
Mies	0,5 %	10,6 %	43,1 %	40,4 %	5,3 %	100,0 %
Nainen	7,4 %	22,2 %	55,6 %	14,8 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	1,4 %	11,9 %	45,4 %	36,7 %	4,6 %	100,0 %

Taulukko 7. Ohjelmoinnin koettu hauskuus sukupuolen mukaan jaoteltuna.

Viimeinen kysymys ennen pelitestauksen aloittamista koski oppilaiden kokonaiskuvaa ohjelmoinnista. Oppilaiden täytyi yksinkertaisesti rastittaa, onko heidän mielikuvansa ohjelmoinnista enemmän positiivinen vai negatiivinen. Tulokset ovat nähtävillä taulukosta 8. Vastausten mukaan 62 % oppilaista näki ohjelmoinnin positiivisena ja 5 % negatiivisena asiana. Neutraalisti ohjelmointiin suhtautui 34 % oppilaista. Sukupuolten välillä oli kuitenkin jälleen merkittäviä eroja vastauksissa. Positiivisena ohjelmoinnin koki 67 % pojista ja 14 % tytöistä. Vastaavasti negatiivisena ohjelmoinnin koki 21 % tytöistä ja 3 % pojista. Kaiken kaikkiaan tytöt suhtautuivat melko neutraalisti ohjelmointiin, kun taas pojista ylivoimainen enemmistö koki ohjelmoinnin positiivisena asiana.

Sukupuoli	Mielikuvani ohjelmoinnista on					Yhteensä
	1. Negatiivinen	2	3	4	5. Positiivinen	
Mies	0,0 %	2,8 %	30,3 %	44,0 %	22,9 %	100,0 %
Nainen	7,1 %	14,3 %	64,3 %	14,3 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	0,0 %	100,0 %	0,0 %	100,0 %
Yhteensä	0,8 %	4,0 %	33,6 %	41,6 %	20,0 %	100,0 %

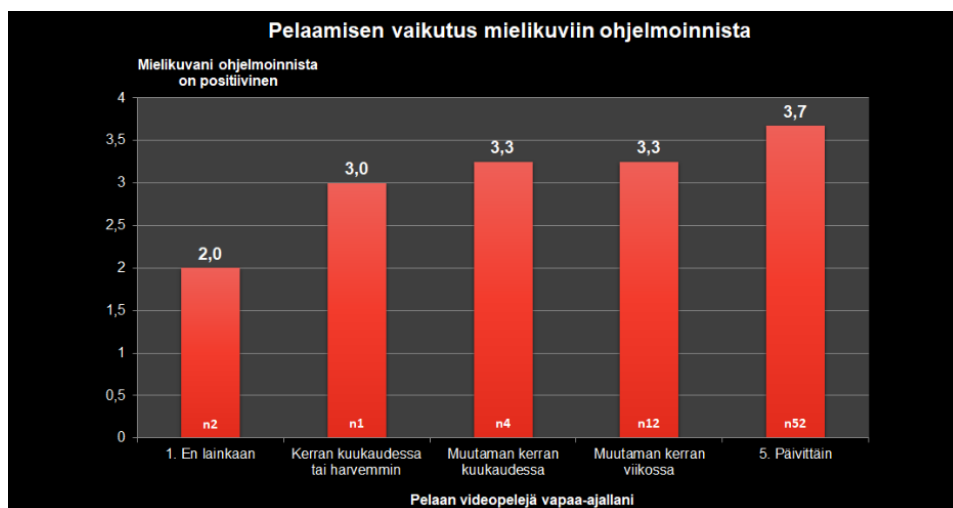
Taulukko 8. Vastaajien mielikuva ohjelmoinnista sukupuolen mukaan jaoteltuna.

4.3 Oppilaan taustan vaikutus näkemyksiin ohjelmoinnista

Sukupuolen lisäksi on kiinnostavaa tarkastella, mitkä muut taustamuuttujat vaikuttavat oppilaiden mielikuviin ohjelmoinnista. Kyselytutkimuksen hypoteesina oli, että mitä enemmän oppilas on ollut vuorovaikutuksessa ohjelmointipelien kanssa, sitä positiivisemmin tämä suhtautuu ohjelmointiin. Tutkimusaineistoa tarkastelemalla havaitaan kuitenkin nopeasti, että tässä tutkimuksessa kartoitetuista taustamuuttujista vain hyvin harva korreloi johdonmukaisesti oppilaan ohjelmointimielikuvien kanssa. Esimerkiksi ohjelmoinnin opetuspelien pelaamisen määrällä oli yllättävän vähän vaikutusta oppilaan mielikuviin ohjelmoinnista. Yli kymmenen tuntia ohjelmoinnin opetuspelejä pelanneet oppilaat kokivat ohjelmoinnin vain lievästi hauskemaksi ja helpommaksi kuin ei lainkaan ohjelmoinnin opetuspelejä pelanneet oppilaat. Samoin ohjelmoinnin opetuspelien pelaaminen ei johdonmukaisesti lisännyt ohjelmoinnin koettua positiivisuutta tai itselle sopivuutta. Ohjelmoinnin opetuspeleillä näyttäisi siis olevan vain vähän pitkäaikaisia vaikutuksia oppilaiden ohjelmointimielikuviin. Tulos on ristiriidassa aiemman tutkimus-

näytön kanssa, jonka mukaan ohjelmoinnin opetuspelien pelaaminen lisää oppilaiden kiinnostusta ja myönteisyyttä ohjelmointia kohtaan.

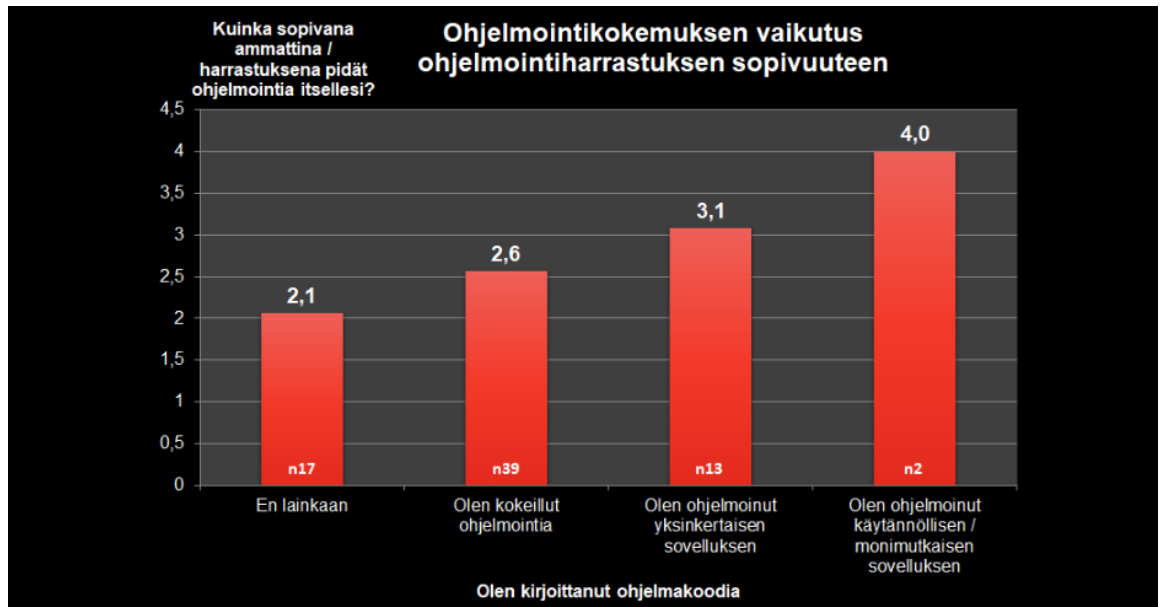
Kerätyistä taustamuuttujista selkeimmin ohjelmointimielikuvien positiivisuuden kanssa korreloivat oppilaan yleinen videopeliharrastuneisuus sekä ohjelmakoodin kirjoittamisen määrä. Tutkimusaineisto paljastaa, että minkä tahansa videopelin säännöllinen pelaaminen lisää ohjelmoinnin koettua positiivisuutta ja hauskuutta. Kuvasta 6 on nähtävillä, että mitä useammin oppilas pelasi videopelejä, sitä positiivisempaan tämä koki ohjelmoinnin. Trendin voidaan päätellä perustuvan siihen, että mitä enemmän oppilas on vuorovaikutuksessa videopelien parissa, sitä kiinnostavampaan tämä kokee videopelien toiminnan ja siten ohjelmoinnin. Pelkästään ohjelmoinnin opetuspelien käyttö opetuksessa ei riitä herättämään samanlaista sisäsyntyistä mielenkiintoa.



Kuva 6. Oppilaat suhtautuivat sitä positiivisemmin ohjelmointiin mitä useammin he pelasivat videopelejä.

Toinen selkeä positiivinen korrelaatio on nähtävillä ohjelmointikokemuksen määrän sekä ohjelmointiharrastuksen koetun sopivuuden välillä. Kuvasta 7 nähdään, että mitä enemmän oppilas on kirjoittanut ohjelmakoodia, sitä sopivampana ammattina tai harrastuksena tämä pitää ohjelmointia itselleen. On kuitenkin muistettava, ettei korrelaatiosta voida päätellä riippuvuussuhteen suuntaa. Luonnollisesti oppilaan kiinnostus ohjelmointiharrastusta kohtaan lisää oppilaan taipumusta kartuttaa ohjelmointikokemusta. Kyse on todennäköisesti itseään ruokkivasta kehästä, jossa kiinnostus ohjelmointia kohtaan lisää ohjelmointikokemusta ja ohjelmointikokemus lisää puolestaan kiinnostusta. Negatiivisin näkemys ohjelmointiharrastuksen sopivuudesta itsel-

le on niillä oppilailla, jotka eivät ole lainkaan kokeilleet ohjelmointia. Positiivisen mielikuvan luominen voi vaatia ensikosketuksen luomista ohjelmointiin, missä opetuspeleillä voi olla merkittävä rooli.



Kuva 7. Ohjelmointikokemus korreloi positiivisesti ohjelmointiharrastuksen koetun sopivuuden kanssa.

Edellä kuvattujen tulosten perusteella nuorten ohjelmointimielikuviin vaikuttaa ennen kaikkea sisäsyntyisen kiinnostuksen määrä. Säännöllinen videopeliharrastus saa ohjelmoinnin tuntumaan kiinnostavalta ja tärkeältä taidolta, mutta toisaalta se ei vähennä mielikuvaa ohjelmoinnin vaikeudesta. Pelkkä videopelien pelaaminen ei myöskään saa ohjelmointia tuntumaan erityisen hauskalta tai itselle sopivalta harrastukselta. Ohjelmointiharrastuksen koettuun sopivuuteen on voimakkaimmin yhteydessä ohjelmointikokemuksen määrä. Lisääntyvä ohjelmointikokemus parantaa kuitenkin vain marginaalisesti ohjelmoinnin koettua helppoutta, hauskuutta ja positiivisuutta. Tulosten perusteella lisääntyvä ohjelmointikokemus näyttää siis lähinnä kasvattavan oppilaan itsevarmuutta omia ohjelmointitaitoja kohtaan, mutta se ei merkittävästi muuta oppilaan mielikuvia ohjelmoinnista. Trendi kertoo oppilaiden mielikuvien muuttamisen haasteellisuudesta. Alkukartoituksen perusteella pelkkä kokemus ohjelmoinnin opetuspeleistä ei riitä luomaan positiivisia ohjelmointimielikuvia tai kiinnostusta ohjelmointiin.

4.4 Mielipiteet DreamScript-opetuspelistä

Oppilaat pelasivat DreamScriptiä noin 50 minuuttia, minkä jälkeen he vastasivat kyselytutkimuksen toiseen vaiheeseen. Tässä vaiheessa oppilaat pystyivät kertomaan pelikokemuksestaan ja antamaan palautetta pelistä. Saatujen vastausten perusteella pelikokemuksen laadussa ei ollut juurikaan eroa tyttöjen ja poikien välillä. Taulukosta 9 nähdään, että sekä tytöistä että pojista 73 % arvioi pelikokemuksensa joko melko tai erittäin hauskaksi. Ainoastaan viidellä prosentilla pelaajista ei ollut hauskaa pelin parissa. Tulosten perusteella DreamScriptin pelaaminen on siis viihdyttävä ja nuorille luonteva tapa tutustua ohjelmoinnin konsepteihin. Kyselyn avoimissa vastauksissa DreamScriptin hauskoiksi puoliksi mainitaan muun muassa ongelmanratkonta, vaihtelevat ympäristöt sekä monipuoliset mahdollisuudet vaikuttaa pelimaailmaan. Ikäviksi puoliksi puolestaan kerrotaan liian vaikeat tehtävät, englannin kieli sekä kohdat, joissa ei tiennyt, mitä tehdä.

Miten hauskaa sinulla oli pelin parissa?						
Sukupuoli	1. Ei lainkaan hauskaa	2	3	4	5. Erittäin hauskaa	Yhteensä
Mies	3,4 %	1,7 %	22,0 %	44,1 %	28,8 %	100,0 %
Nainen	0,0 %	0,0 %	27,3 %	45,5 %	27,3 %	100,0 %
Tyhjä	0,0 %	0,0 %	0,0 %	100,0 %	0,0 %	100,0 %
Yhteensä	2,8 %	1,4 %	22,5 %	45,1 %	28,2 %	100,0 %

Taulukko 9. Vastaajien kokemus DreamScriptin hauskuudesta sukupuolen mukaan jaoteltuna.

Oppilaita pyydettiin myös raportoimaan ne tilanteet, joissa peli ei toiminut odotetulla tavalla. Vastausten perusteella DreamScriptin yleisimmät bugit eli ohjelmistovirheet koskivat pelimaailman fysiikoita sekä pelihahmon liikkumista. Esimerkiksi osa tavaroista saattoi kadota pelimaailmasta, törmäillä miten sattuu tai kulkeutua seinien läpi. Pelihahmo puolestaan saattoi jäädä jumiin maan sisään tai tiputtaa tavaroita itsestään. Bugeja ei kuitenkaan esiintynyt kaikilla pelitestaajilla. Kaiken kaikkiaan 75 % pelaajista ilmoitti DreamScriptin toimineen joko täysin tai melko virheettömästi. Tähän tulokseen voidaan olla ainakin suhteellisen tyytyväisiä, sillä pelitestatun versio oli vielä pitkälti keskeneräinen. Taulukosta 10 nähdään, ettei yksikään pelaajista kokenut DreamScriptiä täysin pelikelvottomaksi.

Miten virheettömästi / bugittomasti peli toimi?						
Sukupuoli	1. Peli oli täysin pelikelvoton	2	3	4	5. Peli toimi täysin virheettömästi	Yhteensä
Mies	0,0 %	3,4 %	22,0 %	45,8 %	28,8 %	100,0 %
Nainen	0,0 %	0,0 %	27,3 %	45,5 %	27,3 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	0,0 %	2,8 %	23,9 %	45,1 %	28,2 %	100,0 %

Taulukko 10. Pelitestaajien havaitsemien bugien määrä sukupuolen mukaan jaoteltuna.

Virheellinen ohjelmakoodi ja niiden aiheuttamat bugit eivät kuitenkaan ole ainoa asia, mikä turhauttaa videopeleissä. Oppilaita pyydettiin raportoimaan kaikki DreamScriptissä turhautumista aiheuttaneet asiat. Avointen vastausten perusteella eritoten vaikeat tehtävät ja niiden synnyttämät epäonnistumisen kokemukset turhauttivat pelitestaajia. Monen vastaajan mielestä peli oli varsinkin aluksi haastava, mutta se helpottui sitä mukaa, kun ohjelmoinnin käsitteet tulivat tutummiksi. Taulukosta 11 nähdään, että 75 % oppilaista ei kokenut DreamScriptiä erityisen turhauttavana. Sen sijaan turhauttavana tai erittäin turhauttavana DreamScriptin koki vain 7 % pelitestaajista. Koska harva piti peliä turhauttavana, sitä voidaan pitää erinomaisena tuloksena ongelmanratkontaa painottavalle ohjelmoinnin opetuspelille. Vastaukset ovat myös yhtenevät tyttöjen ja poikien välillä.

Miten turhauttavana koit pelin?						
Sukupuoli	1. En lainkaan turhauttavana	2	3	4	5. Erittäin turhauttavana	Yhteensä
Mies	28,8 %	47,5 %	16,9 %	6,8 %	0,0 %	100,0 %
Nainen	27,3 %	36,4 %	27,3 %	0,0 %	9,1 %	100,0 %
Tyhjä	0,0 %	100,0 %	0,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	28,2 %	46,5 %	18,3 %	5,6 %	1,4 %	100,0 %

Taulukko 11. DreamScriptin koettu turhauttavuus sukupuolen mukaan jaoteltuna.

Yksi pelikehityksen keskeisimpiä haasteita on pelin vaikeusasteen optimointi. Vaikeusasteen merkitys korostuu varsinkin DreamScriptin kaltaisissa peleissä, jotka on suunnattu mahdollisimman laajalle ja monitaustaiselle yleisölle. Taulukosta 12 nähdään, mitä mieltä yläkoululaiset olivat DreamScriptin vaikeusasteesta. Kaiken kaikkiaan 72 % oppilaista koki pelin vaikeusasteen sopivaksi. Turhan vaikeaksi pelin ilmoitti 18 % oppilaista ja turhan helpoksi 10 %. Tytöt olivat hiukan tyytyväisempiä pelin vaikeustasoon kuin pojat. Tulosten perusteella DreamScriptin pulmia tulisi enemmän helpottaa kuin vaikeuttaa. Pelitestauksen jälkeen DreamScriptiin lisättiin ohjeita ja osa vaikeammista aihealueista siirrettiin myöhempään vaiheeseen peliä. Pelitestaus on äärimmäisen tärkeä työkalu pelin vaikeusasteen kehittämisessä. DreamScriptin saamat tulokset ovat kuitenkin kannustavia, varsinkin kun suurimmalla osalla oppilaista oli hyvin vähän ohjelmointikokemusta taustalla.

Määrittele pelin vaikeusaste						
Sukupuoli	1. Liian helppo	2	3	4	5. Liian vaikea	Yhteensä
Mies	1,7 %	8,5 %	69,5 %	20,3 %	0,0 %	100,0 %
Nainen	0,0 %	9,1 %	81,8 %	9,1 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	1,4 %	8,5 %	71,8 %	18,3 %	0,0 %	100,0 %

Taulukko 12. DreamScriptin koettu vaikeus sukupuolen mukaan jaoteltuna.

Mutta miten hyvin DreamScript toimi oppimisympäristönä? Taulukosta 13 nähdään, miten pelitestaajat arvioivat omaa oppimiskokemustaan. Tällä kertaa tulokset eivät olleet niin selkeitä. Kun oppilailta kysyttiin, lisäkö DreamScript heidän ymmärrystään ohjelmoinnista, 44 % oppilaista oli samaa mieltä ja 21 % eri mieltä väitteen kanssa. Kaiken kaikkiaan 35 % pelaajista ei osannut sanoa, lisäkö peli heidän ymmärrystään ohjelmoinnista vai ei. Tulokset olivat odotettua heikommat, varsinkin kun DreamScriptin vaikeusaste oli valtaosalle oppilaista sopivalla tasolla. Voi olla, että vaikka DreamScriptissä eteneminen vaatii ohjelmointitaitoja, oppilaat eivät kokeneet ymmärryksensä lisääntyneen käsitteellisellä tasolla. Avoimissa vastauksissa kerrotaan etenkin ohjelmoinnin syntaksin tulleen tutummaksi. Moni oppilaista ei kuitenkaan osannut sanoittaa, mitä tarkalleen ottaen oppi pelin aikana. Vähemmän mairittelevista tuloksista huolimatta DreamScriptiä ei tarvitse pitää epäonnistuneena oppimisympäristönä. Tämä käy ilmi etenkin seuraavassa alaluvussa, jossa tarkastellaan miten DreamScript vaikutti oppilaiden mielikuviin ohjelmoinnista.

Sukupuoli	Peli lisäsi ymmärrystäni ohjelmoinnista					Yhteensä
	1. Täysin eri mieltä	2	3	4	5. Täysin samaa mieltä	
Mies	6,8 %	11,9 %	37,3 %	32,2 %	11,9 %	100,0 %
Nainen	0,0 %	36,4 %	27,3 %	36,4 %	0,0 %	100,0 %
Tyhjä	0,0 %	0,0 %	0,0 %	100,0 %	0,0 %	100,0 %
Yhteensä	5,6 %	15,5 %	35,2 %	33,8 %	9,9 %	100,0 %

Taulukko 13. Pelitestaajien kokemus omasta oppimisestaan sukupuolen mukaan jaoteltuna.

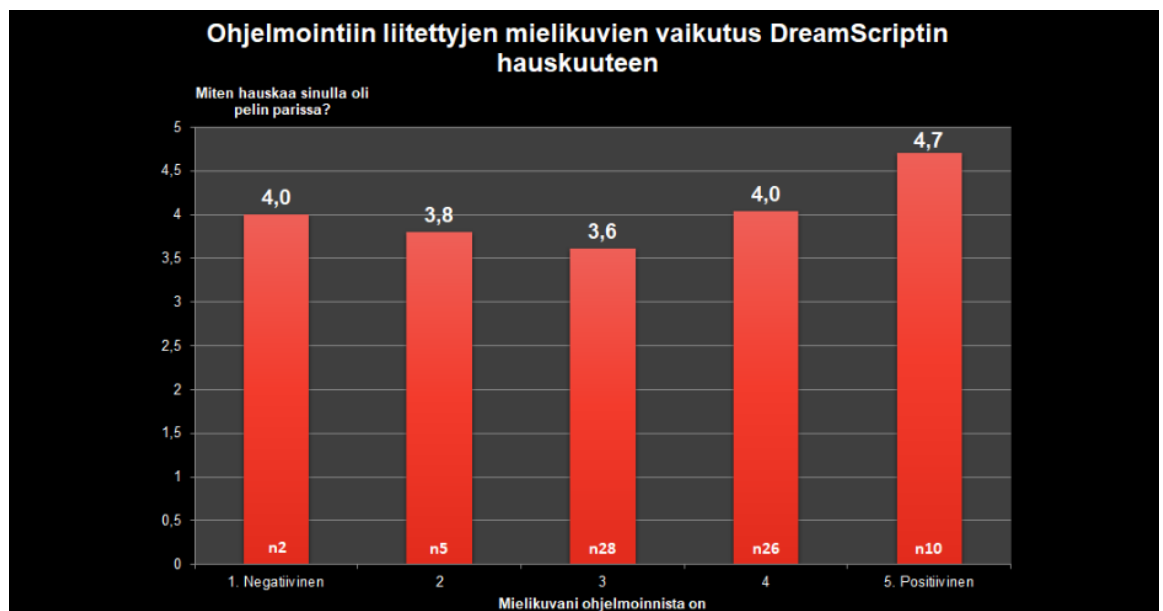
Viimeisenä oppilaiden pelikokemusta käsittelevänä kysymyksenä selvitettiin, kuinka suuri osa oppilaista haluaisi jatkaa DreamScriptin pelaamista myöhemmin. Tulokset ovat nähtävillä taulukosta 14. Kaiken kaikkiaan 41 % oppilaista haluaisi jatkaa pelin pelaamista myöhemmin, kun taas 24 % sai DreamScriptistä tarpeekseen ensimmäisellä kerralla. Samoin 35 % oppilaista ei ollut muodostanut mielipidettä asiasta suuntaan tai toiseen. Tuloksista siis nähdään, että vaikka ylivoimaisella enemmistöllä oli hauskaa DreamScriptiä pelatessaan, se ei vielä riitä kaikkien kouluttamiseen. Toki on huomattava, etteivät ohjelmoinnin opetuspelit ole yläkoululaisten suosikkigenre, eikä kysymyksessä tarkennettu, missä yhteydessä pelin pelaamista jatkettaisiin. Voi olla, että DreamScript on edelleen yläkoululaisille mielenkiintoinen vaihtoehto oppimisympäristöksi, vaikkei se voitakaan vapaa-ajan puhtaasti viihteellisiä videopelejä.

Sukupuoli	Haluaisin jatkaa pelin pelaamista myöhemmin					Yhteensä
	1. Täysin eri mieltä	2	3	4	5. Täysin samaa mieltä	
Mies	5,1 %	16,9 %	33,9 %	27,1 %	16,9 %	100,0 %
Nainen	9,1 %	27,3 %	36,4 %	9,1 %	18,2 %	100,0 %
Tyhjä	0,0 %	0,0 %	100,0 %	0,0 %	0,0 %	100,0 %
Yhteensä	5,6 %	18,3 %	35,2 %	23,9 %	16,9 %	100,0 %

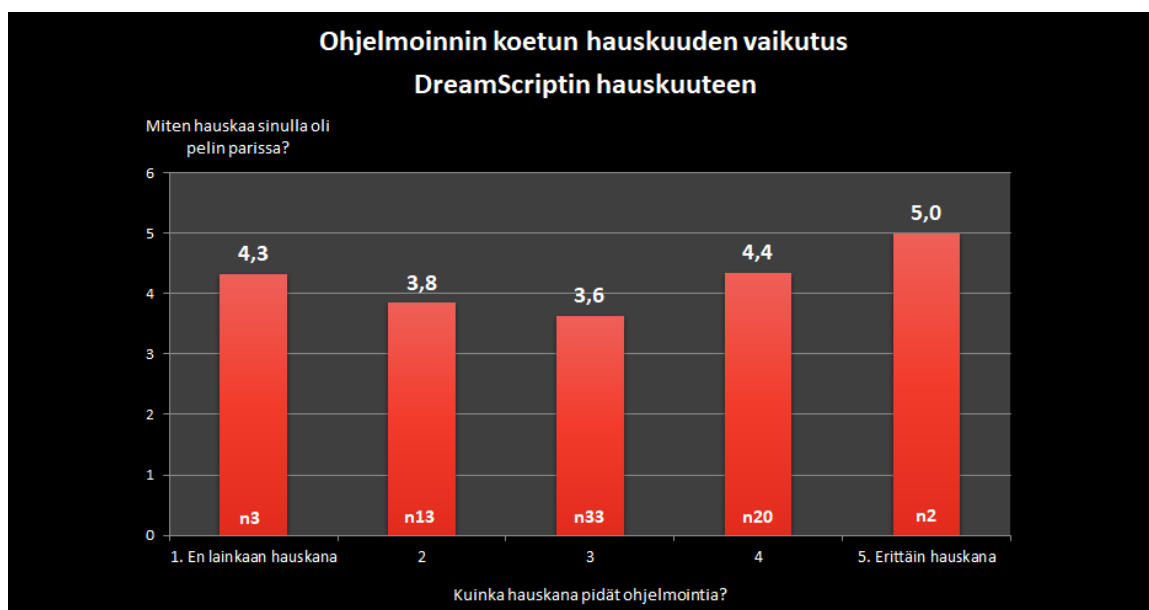
Taulukko 14. Oppilaiden halu jatkaa pelin pelaamista sukupuolen mukaan jaoteltuna.

4.5 Oppilaan taustan vaikutus pelikokemukseen

Tarkastellaan seuraavaksi, miten oppilaiden ilmoittamat taustamuuttujat vaikuttivat DreamScriptin parissa vietettyyn aikaan. Hiukan yllättäen oppilaiden asenteet ja mielikuvat ohjelmoinnista eivät juuri vaikuttaneet oppilaiden kykyyn nauttia DreamScriptin ohjelmointipulmista. Tulos oli yhdenmukainen lukuisia eri mittareita tarkastelemalla. Kuvasta 8 nähdään, että oppilailla oli lähestulkoon yhtä hauskaa DreamScriptin parissa riippumatta siitä, oliko heidän yleinen suhtautumisensa ohjelmointiin positiivinen tai negatiivinen. Samoin kuvasta 9 nähdään, ettei ohjelmoinnin koettu hauskuus vaikuttanut merkittävästi DreamScriptin koettuun hauskuuteen. Luonnollisesti ne oppilaat, jotka kokivat ohjelmoinnin kaikkein hauskimpana, kokivat myös DreamScriptin kaikkein hauskimpana. Kuitenkin DreamScript sai ohjelmoinnin tuntumaan hauskalta myös niiden oppilaiden keskuudessa, jotka eivät normaalisti pidä ohjelmointia hauskana harrasteena. Ainoastaan ne oppilaat, joilla ei ollut voimakasta mielipidettä sen enempää ohjelmoinnin positiivisuudesta tai hauskuudesta, ei ollut voimakasta mielipidettä myöskään DreamScriptin hauskuudesta.

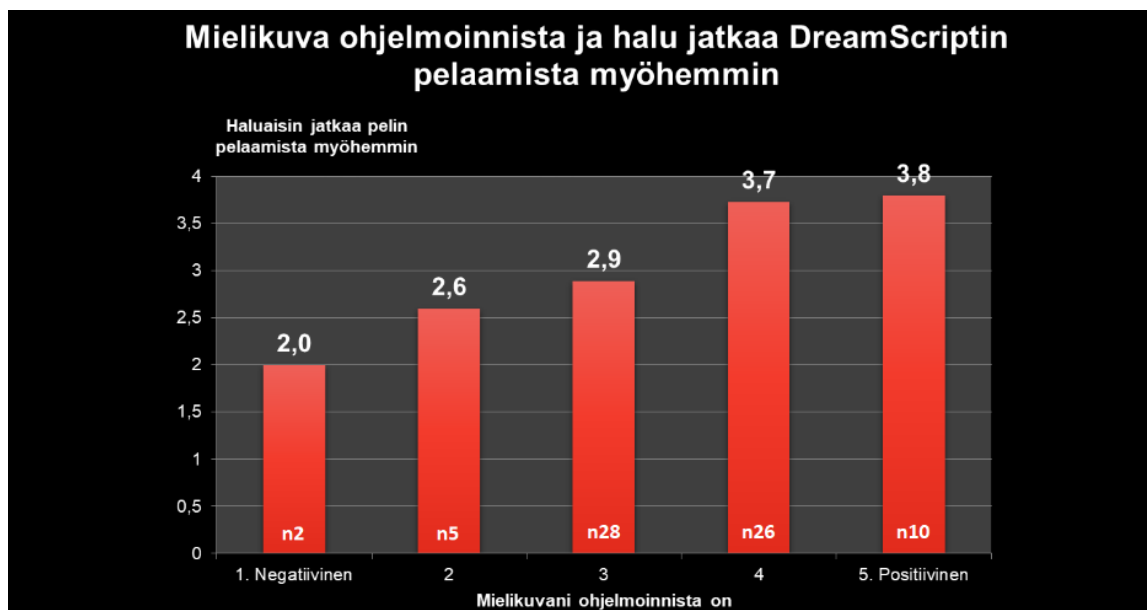


Kuva 8. Ohjelmointiin liitetyt mielikuvat eivät vaikuttaneet merkittävästi oppilaiden kykyyn nauttia DreamScriptin pelaamisesta.



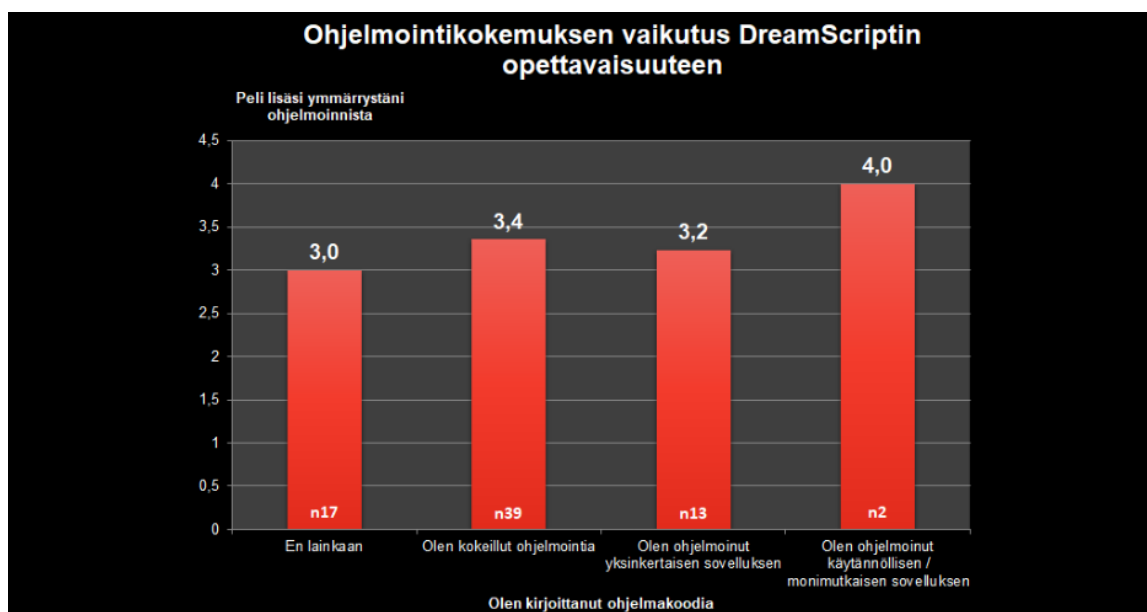
Kuva 9. Oppilailla oli hauskaa DreamScriptin parissa riippumatta siitä, miten hauskana he kokivat yleisesti ohjelmoinnin.

Edellä kuvattu trendi kertoo virtuaalimaailman ja interaktiivisuuden voimasta kääntää ikävätkin aiheet mielekkäiksi oppimisympäristöiksi. Toisaalta tulos kertoo myös siitä, etteivät ohjelmointiin liitetyt negatiiviset mielikuvat aina perustu omakohtaisiin kokemuksiin ohjelmoinnista. Tulosten perusteella DreamScript saa yhä useamman oppilaan nauttimaan ohjelmointitehtävistä. Tämä ei kuitenkaan tarkoita, että DreamScript tai mikään muukaan ohjelmoinnin opetuspelejä yksistään muuttaisi oppilaiden motivaation sisäsyntyiseksi. Vaikka videopelit voivat madaltaa tiettyyn asiaan tutustumisen kynnyksen, oppilaiden perimmäisiin kiinnostuksen kohteisiin niillä ei välttämättä ole pitkäaikaisia vaikutuksia. Esimerkiksi kuvasta 10 nähdään, että mitä positiivisemmin oppilas suhtautui ohjelmointiin, sitä suuremmalla todennäköisyydellä tämä halusi jatkaa DreamScriptin pelaamista myöhemmin. Ennakkoasenteilla on siis varsin pysyviä vaikutuksia oppilaan toimintaan ja motivaatioon, vaikka yksittäinen pelisessio olisikin hauska DreamScriptin parissa.



Kuva 10. Ohjelmointiin liitetyt mielikuvat vaikuttavat voimakkaasti siihen, haluaisiko oppilas jatkaa DreamScriptin pelaamista myöhemmin.

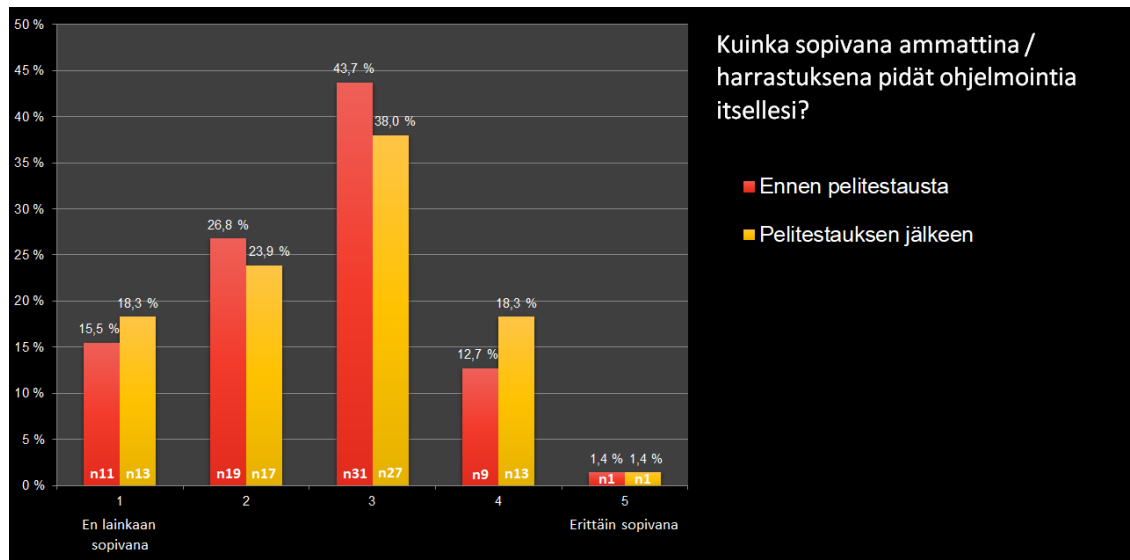
Tarkastellaan seuraavaksi, miten oppilaiden ohjelmointikokemuksen määrä vaikutti DreamScriptin toimivuuteen opetuspelinä. Hypoteesina oli, että mitä vähemmän oppilaalla on ohjelmointikokemusta, sitä hyödyllisempänä tämä kokisi DreamScriptin pelaamisen. Saadut tulokset eivät kuitenkaan tukeneet tätä hypoteesia. Kuvasta 11 nähdään, että itse asiassa ne oppilaat, joille oli kertynyt kaikkein eniten ohjelmointikokemusta, kokivat myös saavansa eniten hyötyä DreamScriptin pelaamisesta. Vastaavasti vähiten hyötyä DreamScriptin pelaamisesta kokivat saavansa ne oppilaat, joilla ei ollut lainkaan ohjelmointikokemusta. Vaikka erot eivät ole suuria ja oppilaiden kokemuskajuma oli hyvin epätasainen, kuvastaa tulos kuitenkin hyvin ohjelmoinnin luonnetta abstraktina ongelmanratkaisuna. Ohjelmoinnin opetuksessa alkutaival on yleensä kaikkein haastavin. Vasta kun ohjelmoinnin ajattelumallista on saanut kokemuksen myötä selkeämmän käsityksen, on olemassa olevan tiedon päälle helppo rakentaa uutta. Edellä kuvattu konstruktivistinen oppimismalli saattaa näkyä tuloksissa esimerkiksi siten, että vain vähän ohjelmointikokemusta omaavilla oppilailla pelikokemus jätti niin paljon avoimia kysymyksiä, että heidän oli vaikea arvioida saamaansa hyötyä pelistä. Runsaasti ohjelmointikokemusta omaavat oppilaat puolestaan saivat DreamScriptin pelaamisen kautta kätevän kertauksen siitä, miten ohjelmoinnin syntaksi ja komennot toimivat. Näin epätasaisella kokemuskajumalla tuloksista ei kuitenkaan voida tehdä pitkälle vietyjä päätelmiä.



Kuva 11. Pelaajat, joilla oli eniten ohjelmointikokemusta, kokivat myös hyötyvänsä eniten DreamScriptin pelaamisesta.

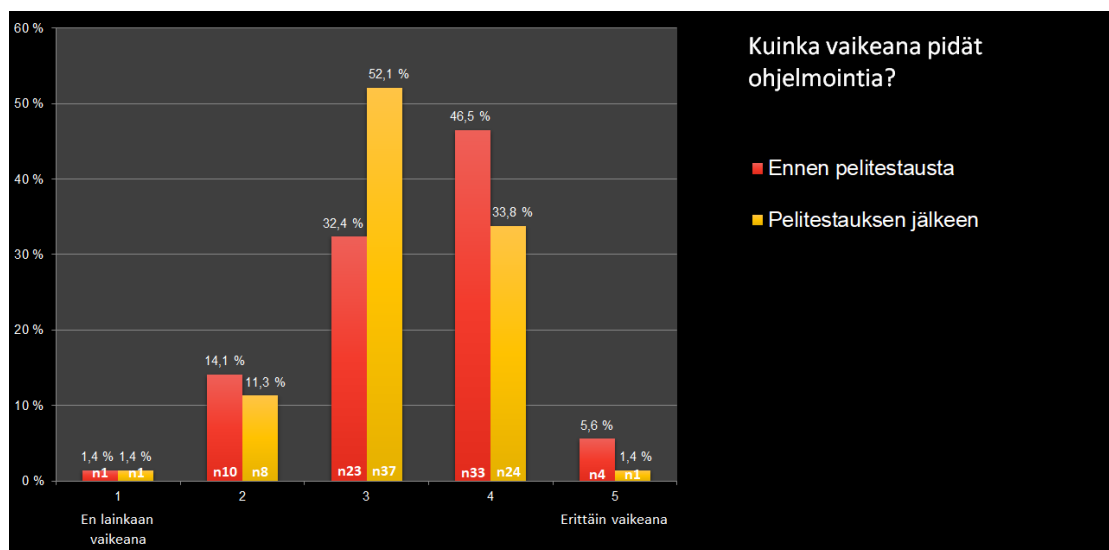
4.6 Mielikuvat ohjelmoinnista pelitestauksen jälkeen

Pelitestaussektion jälkeen oppilaiden mielikuvat ohjelmoinnista kartoitettiin uudelleen. Ensimmäisenä kerrattiin kysymys siitä, kuinka sopivana ammattina tai harrastuksena oppilaat pitävät ohjelmointia itselleen. Kuvasta 12 nähdään, että pelisession myötä karttuneen ohjelmointikokemuksen ansiosta oppilaat pystyivät tarkemmin arvioimaan kiinnostustaan ohjelmointia kohtaan. Pelisession seurauksena sekä ohjelmointiharrastukseen positiivisesti että negatiivisesti suhtautuvien oppilaiden määrä lisääntyi. Muutos oli suurempi positiiviseen kuin negatiiviseen suuntaan. Vaikka yksittäisten vastaajien ajatuksenjuoksua ei voida varmaksi tietää, tuloksia voidaan tulkita niin, ettei DreamScriptin pelaaminen muuttanut voimakkaasti oppilaiden mielipidettä ohjelmointiharrastuksen sopivuudesta. Sen sijaan näkemykset alkoivat hitaasti liikkua kohti jompaakumpaa ääripäätä. Toisin sanoen pelitestauksen jälkeen oppilaat näkivät ohjelmoinnin joko lievästi mielenkiintoisempana tai lievästi epäsovivampana harrastuksena kuin ennen pelitestausta.



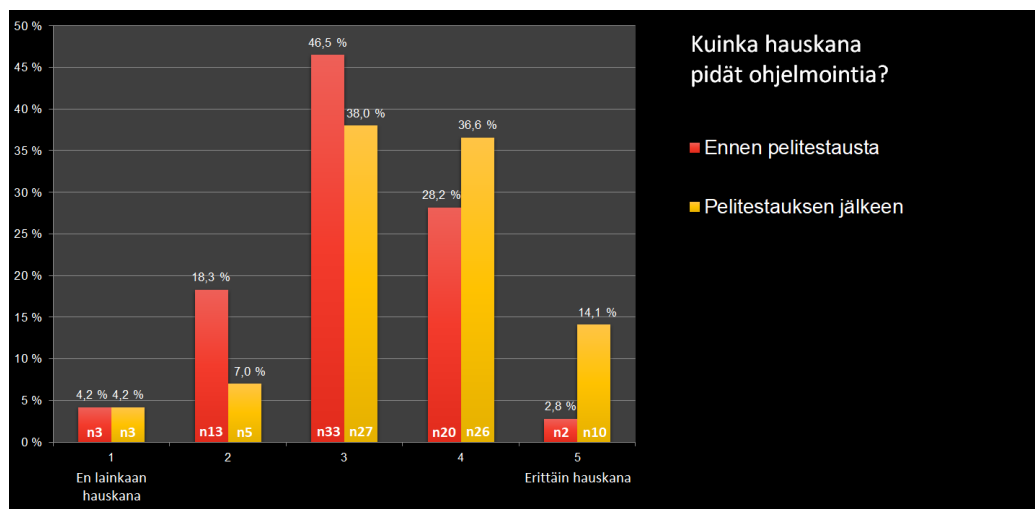
Kuva 12. Kiinnostus ohjelmointiharrastusta kohtaan ennen ja jälkeen pelitestauksen.

Kuva 13 näyttää miten oppilaiden mielikuva ohjelmoinnin vaikeudesta muuttui DreamScriptin pelaamisen myötä. Tulosten mukaan ohjelmointi nähtiin merkittävästi helpompana opetuspelellä pelaamisen jälkeen. Ennen pelitestausta ohjelmoinnin koki erittäin tai joihinkin vaikeaksi 52 % oppilaista. Pelitestauksen jälkeen vastaava luku oli enää 35 %. Tästä voidaan päätellä DreamScriptin toimineen tarkoituksenmukaisella tavalla. Peli tarjosi oppilaille mahdollisuuden kokeilla ohjelmoinnin perusteita hausassa ja turvallisessa ympäristössä, mikä sai ohjelmoinnin näyttävämmän heille vähemmän haastavana. Pelitestauksen jälkeen neutraalisti ohjelmoinnin vaikeuteen suhtautuvien oppilaiden määrä lisääntyi jopa 61 %. Tulos kertoo oppilaiden yllättyneen ohjelmoinnin helppoudesta. Sen sijaan ohjelmoinnin jo lähtökohtaisesti helppona kokeneiden oppilaiden määrä ei pelitestauksen myötä juuri muuttunut.



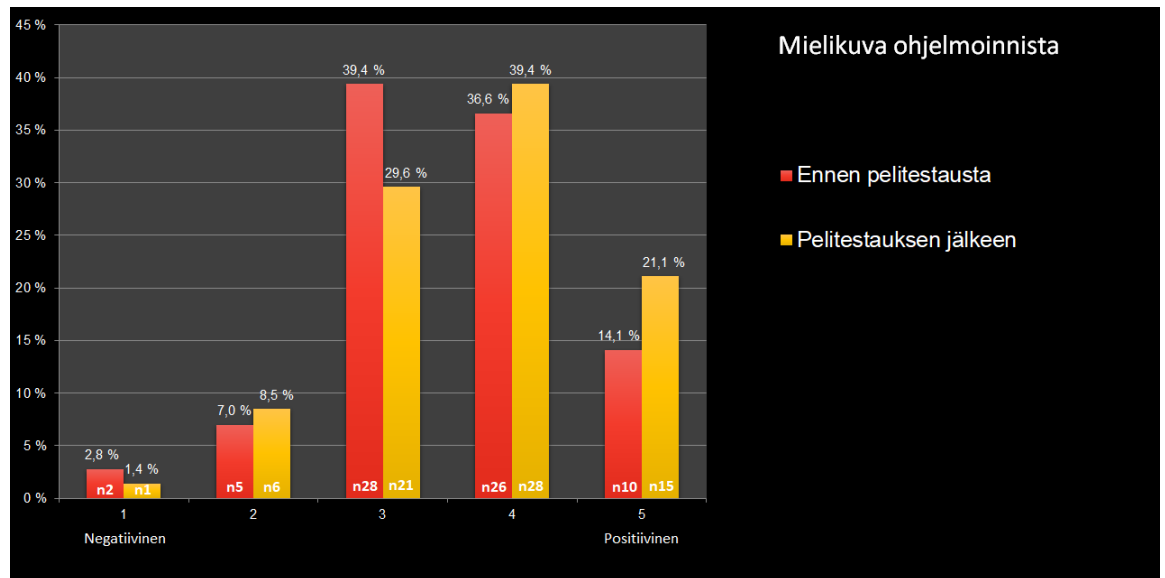
Kuva 13. Ohjelmoinnin koettu vaikeus ennen ja jälkeen pelitestauksen.

DreamScriptin pelaamisella oli kaikkein suurin vaikutus oppilaiden mielikuvaan ohjelmoinnin hauskuudesta. Kuvasta 14 nähdään, että ennen pelitestausta ohjelmoinnin koki erittäin tai jokin hauskaksi 31 % oppilaista. Pelitestauksen jälkeen vastaava luku oli jo 51 %. Vastaavasti ohjelmoinnin tylsänä kokeneiden oppilaiden määrä putosi DreamScriptin pelaamisen myötä 23 prosentista 11 prosenttiin. Voidaan siis todeta, että DreamScript sai ohjelmoinnin tuntumaan oppilaista sekä hauskemmalta että helpommalta. Seuraavaksi tarkastellaan, miten nämä positiiviset muutokset mielikuvissa näkyvät oppilaiden kokonaisvaltaisessa asenteessa ohjelmointia kohtaan.



Kuva 14. Ohjelmoinnin koettu hauskuus ennen ja jälkeen pelitestauksen.

Pelitestauksen jälkeen oppilaita pyydettiin jälleen vastaamaan, onko ohjelmointi heistä enemmän positiivinen vai negatiivinen asia. Oppilaiden asennemuutokset ovat nähtävillä kuvasta 15. Ennen pelitestausta oppilaista 51 % koki ohjelmoinnin positiivisena ja 10 % negatiivisena asiana. Pelitestauksen jälkeen vastaavat luvut olivat 61 % ja 10 %. DreamScriptin pelaaminen sai siis oppilaat näkemään ohjelmoinnin aiempaa positiivisempänä asiana, mutta se ei onnistunut vähentämään ohjelmointiin negatiivisesti suhtautuvien määrää. Kerran luotujen negatiivisten mielikuvien muuttaminen näyttäisi siis olevan erityisen haastavaa. Mutta toisaalta kuvasta 15 nähdään, että kaikkein myönteisimmin ohjelmointiin suhtautuneiden oppilaiden määrä lisääntyi DreamScriptin pelaamisen myötä eniten, mikä on lupaava tulos.



Kuva 15. Mielikuva ohjelmoinnista ennen ja jälkeen pelitestauksen.

Yhteenvetona voidaan todeta, että DreamScriptin pelaaminen lisäsi ohjelmointia hauskana pitävien oppilaiden määrää 64 %. Samoin oppilaiden, jotka kokivat ohjelmoinnin itselle sopivaksi harrastukseksi, määrä kasvoi 40 %. Lisäksi ohjelmoinnin positiivisena näkevien oppilaiden määrä lisääntyi 19 %. Ohjelmointia vaikeana pitävien oppilaiden määrä puolestaan väheni 32 %. DreamScriptin vaikutus oppilaiden ohjelmointimielikuviin oli siis kauttaaltaan positiivinen. Muutos on yllättävä varsinkin siitä näkökulmasta, että oppilaista vähemmistö, vain 44 % koki DreamScriptin varsinaisesti lisänneen heidän ymmärrystään ohjelmoinnista. Tulokset ovat mielenkiintoisia myös taustamuuttujien näkökulmasta. Pelitestausta edeltäneen alkukartoituksen perusteella ohjelmoinnin opetuspelien pelaamisen ei pitäisi korreloida oppilaan ohjelmointimielikuvien kanssa. Kuten alaluvussa 4.3 todettiin, kokemus ohjelmoinnin opetuspelien pelaamisesta ei johdonmukaisesti lisännyt oppilaan positiivisia mielikuvia ohjelmoinnin hauskuudesta, helppoudesta, positiivisuudesta tai itselle sopivuudesta. DreamScriptin kohdalla nämä muutokset ovat kuitenkin selkeästi nähtävillä. Onko DreamScript siis poikkeuksellisen tehokas ohjelmoinnin opetuspelejä oppilaiden ohjelmointimielikuvien muuttamiseen? Sitä on näiden tulosten perusteella mahdoton arvioida, koska kysely suoritettiin välittömästi pelitestaussession jälkeen. Se, millaisia pitkäaikaisia vaikutuksia DreamScriptin pelaamisella on oppilaiden ohjelmointimielikuviin jää siis mahdollisten myöhempien tutkimusten selvitettäväksi.

4.7 Tutkimusasetelman kritiikki

DreamScriptin pelitestaukseen ja sen ohessa suoritettuun kyselytutkimukseen liittyy lukuisia epävarmuustekijöitä, jotka vähentävät edellä kuvattujen tulosten luotettavuutta ja niiden yleistettävyyttä. Ensinnäkin osa pelitestaussessioista järjestettiin ATK:n valinnastunneilla, mikä todennäköisesti vaikuttaa oppilaiden ohjelmointimielikuviin ja motivaatioon pelata DreamScriptiä. Keskimääräinen koululainen ei välttämättä ole yhtä orientoitunut tietotekniikan käyttöön opetuksessa kuin ATK:n valinnaisaineeksi ottanut. Toisekseen pelitestaajien sukupuolijakauma oli hyvin epätasainen, mikä voi vääristää tuloksia ja saada sukupuolierot näyttämään todellista suuremmilta. Pelitestaajista 59 oli poikia ja 11 tyttöjä. Yksi vastaajista jätti sukupuolikohdan tyhjäksi. Epätasaisen otoksen vuoksi sukupuolten väliset erot esimerkiksi ohjelmoinnin koetussa vaikeudessa voivat selittyä sattumalla. Ääripään vastaukset saattavat korostua tyttöjen vastauksissa, koska heitä ei ollut tarpeeksi vastaamassa ja siten tasoittamassa tuloksia.

Yksi merkittävimmistä tutkimusasetelman epäkohdista oli kontrolliryhmän puute. Ilman kontrolliryhmää on mahdoton sanoa, johtuivatko muutokset oppilaiden ohjelmointimielikuvissa DreamScriptin pelaamisesta vai puhtaasti ajan kulumisesta. Ihminen ei välttämättä vastaa samalla tavalla samoihin kysymyksiin, vaikka kyselyiden välissä olisi aikaa alle tunti. Oppilaat saattavat olla positiivisemmalla tuulella oppitunnin lopussa kuin alussa, riippumatta siitä, mitä oppitunnilla tehdään. Oppilaat voivat myös kokea sosiaalista painetta vastata kyselyihin tietyllä tavalla. Kyselytutkimuksen vastaajat haluavat usein vastata siten, kuten heidän odotetaan tai toivotaan vastaavan. Taipumuksen taustalla on motiivi näyttäytyä ennemmin myönteisessä kuin täsmällisessä valossa. [45.] Tulokset DreamScriptin toimivuudesta oppimisympäristönä saattavat siis olla positiivisesti vääristyneitä. Mahdollisuus opetuspelin pelaamiseen oppitunnilla voi tuntua jo itsessään niin mieltäsalta, ettei pelikokemusta haluta arvioida kriittisesti.

Kaikki tutkimusasetelman ratkaisuista eivät kuitenkaan olleet DreamScriptin tuloksia parantavia. Esimerkiksi vastaukset kyselytutkimuksen väittämään ”Peli lisäsi ymmärrystäni ohjelmoinnista”, sisältää myös niiden oppilaiden vastaukset, joilla oli jo valmiiksi runsaasti ohjelmointikokemusta. Ainakin yhdessä avoimessa vastauksessa todetaan, ettei DreamScript opettanut mitään, koska pelitestaaja tunsikin jo entuudestaan pelin opettamat ohjelmointiteemat. Toisekseen pelitestauksen suhteellisen lyhyt kesto, viisikymmentä minuuttia, ei välttämättä ollut riittävä aika DreamScriptin pelimekaniikkaan ja ohjelmointiin tutustumiseen. Yksikään pelitestaajista ei ehtinyt pelata DreamScriptiä läpi testausajassa. Kolmanneksen DreamScriptin pelitestattu versio oli vielä hyvin keskeneräinen, eivätkä tulokset vastaa pelin lopullista laatua tai toimivuutta opetuspelinä.

5 DreamScriptin tulevaisuus

DreamScriptin aktiivinen kehitys päättyi joulukuussa 2020. Sen jälkeen peliin on lisätty satunnaisesti uusia päivityksiä. Pelitestaus antoi tärkeää tietoa siitä, miten pelaajat käyttäytyvät pelimaailmassa ja mihin suuntaan DreamScriptiä tulisi jatkokehittää. Tehtyjen havaintojen sekä pelaajilta saadun palautteen perusteella muun muassa DreamScriptin vaikeusastetta on säädetty vastaamaan paremmin yläkouluikäisten taitoja. Vaikeusasteen optimointi toteutettiin pidentämällä pelin tutoriaalia, antamalla pelaajalle täsmällisempiä vinkkejä, muuttamalla kenttäsuunnittelua niin, että pelaajaa kannustetaan tutkimaan aiempaa enemmän ympäristöään sekä tekemällä pelihahmon liikkumisesta ja fysiikkaobjektien käsittelystä sujuvampaa. On mielenkiintoista nähdä, miten tehdyt muutokset vaikuttavat pelikokemuksen laatuun tulevaisuudessa.

DreamScriptin voi tulkita olevan paitsi ohjelmoinnin opetuspelejä, myös osa kokeellisten pulmapelien jatkumoa. Kokeellisten pelien arvo on siinä, että ne tuovat pelejä kokonaan uusille yleisöille. Parhaimmillaan kokeelliset pelit muuttavat pysyvästi sitä, miten pelejä kehitetään jatkossa – onhan jokainen vakiintunut peli-idea ollut kokeellinen omana aikanaan. DreamScriptin kokeellisuus syntyy siitä, että se antaa pelaajan muokata asioita, jotka ovat perinteisesti olleet vain pelin kehittäjien käsissä. Pelaaja voi esimerkiksi luoda uusia olioita tyhjästä, muuttaa kumiankan norsun painoiseksi sekä haihduttaa laatikon kaasuksi ja kuljettaa sen aidan läpi. DreamScript pakottaa ajattelemaan kuin ohjelmoija ja käyttämään ohjelmoinnin abstrakteja käsitteitä konkreettisissa ongelmatilanteissa. Pelin sanoma on, että teknologia on ihmeellistä ja sitä tulee käyttää luoviin ratkaisuihin.

DreamScript on herättänyt kiinnostusta pelialan ammattilaisten, pulmapelien ystävien sekä ohjelmoinnin harrastajien keskuudessa. DreamScript voitti opiskelijoille suunnatun Bit1 2021 -pelinkehityskilpailun. Kilpailussa arvioitiin kymmeniä suomalaisissa korkeakouluissa tuotettuja opiskelijapelejä. Arviointikriteereinä toimivat muun muassa pelin kaupallinen potentiaali, omaperäisyys sekä tekninen laatu. Tuomareiden arvion mukaan DreamScript vetoaa lukuisiin kohderyhmiin sekä onnistuu tekemään oppimisesta hauskaa. Lisäksi tuomarit olivat vaikuttuneita pelin kauniista grafiikasta. [46.] Kilpailumenestyksen sekä positiivisen palautteen myötä DreamScriptin jatkokehitykselle on etsitty rahoitusta niin julkaisijoiden kuin Suomen Kulttuurirahaston taholta. Rahoituksen avulla DreamScriptiin saataisiin lisättyä paljon kaivattuja ominaisuuksia, kuten kenttäeditori, sujuvampi käyttöliittymä sekä kerättäviä saavutuksia. Neuvottelut rahoituksesta ovat edelleen käynnissä ja toistaiseksi varmin tieto DreamScriptin tulevaisuudesta on se, että peli julkaistaan muodossa tai toisessa vuoden 2022 aikana.

6 Yhteenveto

Ohjelmoinnin opetuspelit ovat hyödyllisiä, mutta eivät ongelmattomia oppimisympäristöjä. Ne lisäävät oppilaiden aktiivisuutta, pakottavat soveltamaan tietoa sekä kehittävät päättelykykyä. Haittapuolena ohjelmoinnin opetuspelit saattavat synnyttää väärinkäsityksiä tai tukea haitallisia oppimisstrategioita, jos oppilaalla on vaikeuksia pelin parissa. Käytännössä ohjelmoinnin opetuspelit toimivat parhaiten lähiopetukseen yhdistettynä, jolloin oppilailla on mahdollisuus saada neuvoa ja tukea pelien käytössä. Erityisen hyvin opetuspelit toimivat motivaation herättämisessä uuteen aiheeseen perehdyttäessä tai kun halutaan demonstroida jotain käytännön ilmiötä.

Ohjelmoinnin opetuspelien kehitys vaatii paljon testausta sekä pelin tavoitteita tukevaa suunnittelua. Erityisiä haasteita tuottaa opetuspelin hauskuuden, vaikeusasteen sekä teoriasisällön tasapainottaminen. Ihannetilanteessa ohjelmoinnin opetuspeli on helppo omaksua, mutta se tarjoaa työkalut myös vaativien toimintojen kehittämiseen. Kun pelaajalle annetaan tilaa omaehtoiseen kokeiluun sekä oivaltamisen iloon, syntyy otollinen ympäristö oppilaan sisäsyntyisen motivaation vahvistumiseen. On kuitenkin huomattava, että osa pelaajista tarvitsee enemmän ohjausta ja kaipaa opetuspeliltä selkeitä tavoitteita. Ohjelmoinnin opetuspelien kehityksessä joudutaankin huomioimaan niin erilaisten pelaajien kuin kouluttajien tarpeet. Parhaat ohjelmoinnin opetuspelit onnistuvat minimoimaan pelin käyttöönottokustannukset ja maksimoimaan pelistä saatavat hyödyt.

DreamScript on Kajaanin ammattikorkeakoulussa opiskelijaprojektina kehitetty ohjelmoinnin opetuspeli yli 12-vuotiaille. DreamScriptiä pelitestiin kajaanilaisissa yläkouluissa viidessä eri pelisessiossa syksyllä 2020. Yhteensä kyselytutkimukseen osallistui 71 oppilasta. Tulosten mukaan DreamScriptin pelaaminen sai ohjelmoinnin tuntumaan oppilaista aiempaa hauskemmalta, helpommalta sekä itselle sopivammalta harrastukselta. Myös oppilaiden kokonaisvaltainen suhtautuminen ohjelmointiin muuttui pelin myötä positiivisemmaksi. DreamScript ei kuitenkaan kyennyt vähentämään ohjelmointiin negatiivisesti suhtautuneiden oppilaiden määrää. Tulosten perusteella DreamScript toimii kuitenkin hauskana tapana tutustua ohjelmointiin kannustavassa ja turvallisessa ympäristössä. Tämä siitäkin huolimatta, että vähemmistö, 44 % oppilaista koki DreamScriptin varsinaisesti lisänneen heidän ymmärrystään ohjelmoinnista. DreamScriptin suurimmat ansiot ovatkin sen kyky parantaa oppilaiden itseluottamusta omia ohjelmointitaitoja kohtaan sekä kyky vähentää ohjelmointiin liitettyjä negatiivisia stereotypioita. Saadut tulokset ovat lupaavia ja tukevat aiempien tutkimusten näyttöä ohjelmoinnin opetuspeleistä nuorille luontevana sekä hyödyllisenä oppimisympäristönä.

Lähteet

- (1) Mertala P, Palsa L, Dufva, T. (2020). Monilukutaito koodin purkajana: Ehdotus laaja-alaiseksi ohjelmoinnin pedagogiikaksi. *Media & viestintä*, 43(1), 21–46.
- (2) Uudet lukutaidot -kehittämishjelma. Opetus- ja kulttuuriministeriö. Saatavilla: <https://okm.fi/uudet-lukutaidot>. Noudettu 18.11.2021.
- (3) Opetushallitus. Perusopetuksen opetussuunnitelman perusteet 2014. Helsinki; 2016. Saatavilla: https://www.oph.fi/sites/default/files/documents/perusopetuksen_opetussuunnitelman_perusteet_2014.pdf. Noudettu 18.11.2021.
- (4) Mitgutsch K, Alvarado N. (2012). Purposeful by Design? A Serious Game Design Assessment Framework. *Proceedings of the International Conference on the foundations of digital games*, 121–128.
- (5) Calvo-Morata, A, Alonso-Fernández C, Freire M, Martínez-Ortiz I, Fernández-Manjón B, (2020). Serious games to prevent and detect bullying and cyberbullying: A systematic serious games and literature review. *Computers and education*, 157.
- (6) Chow C, Riantiningtyas R, Kanstrup M, Papavasileiou M, Liem G, Olsen A. (2020). Can games change children's eating behaviour? A review of gamification and serious games. *Food quality and preference*, 80.
- (7) Solinska-Nowak A, Magnuszewski P, Curl M, French A, Keating A, Mochizuki J, Liu W, Mechler R, Kulakowska M, Jarzabek L. (2018). An overview of serious games for disaster risk management – Prospects and limitations for informing actions to arrest increasing risk. *International journal of disaster risk reduction*, 31, 1013–1029.
- (8) Silveira I, Villalba K. (2018). An Open Perspective for Educational Games. *Journal of information technology research*, 11(1), 18–28.
- (9) Risnawati, Amir Z, Wahyuningsih D. (2018). The Development of Educational Game as Instructional Media to Facilitate Students' Capabilities in Mathematical Problem Solving. *2nd International Conference on Statistics, Mathematics, Teaching, and Research*, 1028–1035.

- (10) Ibrahim R, Zairah N, Wong D, C.M R, Maarop N, Yaacob S. (2018). Student's Opinions on Online Educational Games for Learning Programming Introductory. *International Journal of Advanced Computer Science & Applications*, 9(6), 332–338.
- (11) Mathew R, Iqbal S, Tawafak R. (2019). Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. *Informatics in Education*, 18(2), 359–373.
- (12) Mathrani A, Christian S, Ponder-Sutton A. (2016). PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Educational Technology & Society*, 19(2), 5–17. *International Forum of Educational Technology & Society*. Palmerston North.
- (13) Melcer E, Hollis V, Isbister K. (2017). Tangibles vs. Mouse in Educational Programming Games: Influences on Enjoyment and Self-Beliefs.
- (14) Lindberg R, Laine T, Haaranen L. (2019). Gamifying programming education in K-12: A review of programming curricula in seven countries and programming games. *British journal of educational technology*, 50(4), 1979–1995.
- (15) Papadakis S. (2021). The Impact of Coding Apps to Support Young Children in Computational Thinking and Computational Fluency. A Literature Review. *Frontiers in education*, 6, 11–21.
- (16) Rose S, Habgood J, Jay T. (2020). Designing a Programming Game to Improve Children's Procedural Abstraction Skills in Scratch. *Journal of educational computing research*, 58(7), 1372–1411.
- (17) Lode H, Franchi G, Frederiksen N. (2013). Machineers: Playfully Introducing Programming to Children. *Extended abstracts on human factors in computing systems*, 2639–2642.
- (18) Giannakoulas A, Xinogalos S. (2018). A pilot study on the effectiveness and acceptance of an educational game for teaching programming concepts to primary school students. *Education and Information Technologies*, 23, 2029–2052. Springer Science and Business Media LLC. New York.
- (19) Rugelj J, Lapina M. (2019). Game design based learning of programming. *Proceedings of SLET-2019 – International Scientific Conference Innovative Approaches to the Application of Digital Technologies in Education and Research*. Saatavilla: http://ceur-ws.org/Vol-2494/invited_paper_4.pdf. Noudettu 18.11.2021.

- (20) Malliarakis C, Satratzemi M, Xinogalos S. (2014). Educational Games for Teaching Computer Programming. *Research on e-learning and ICT in Education: Technological, Pedagogical and Instructional Perspectives*, 87–98. Springer New York.
- (21) Esper S, Stephen F, Griswold W. (2013). CodeSpells: Embodying the Metaphor of Wizardry for Programming. *Proceedings of the 18th ACM conference on innovation and technology in computer science education*, 249–254.
- (22) Zhu J, Alderfer K, Smith B, Char B, Ontañón S. (2020). Understanding Learners' Problem-Solving Strategies in Concurrent and Parallel Programming: A Game-Based Approach. *Saatavilla*: <https://arxiv.org/pdf/2005.04789.pdf>. Noudettu 18.11.2021.
- (23) Vihavainen A, Airaksinen J, Watson C. (2014). A systematic review of approaches for teaching introductory programming and their influence on success. *Proceedings of the tenth annual conference on international computing education research*, 19–26.
- (24) Malliarakis C, Satratzemi M, Xinogalos S. (2014). Designing Educational Games for Computer Programming: A holistic Framework. *The Electronic Journal of e-Learning*, 12(3), 281–297.
- (25) Villareale J, Biemer C, El-Nasr M, Zhu J. (2020). Reflection in Game-Based Learning: A Survey of Programming Games, 1–13.
- (26) De Freitas S. (2018). Are Games Effective Learning Tools? A Review of Educational Games. *Educational Technology & Society*, 21(2), 80.
- (27) Repenning A, Webb D, Ioannidou A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. *ACM*, 265–269.
- (28) Xinogalos S, Tryfou M. (2021). Using Greenfoot as a Tool for Serious Games Programming Education and Development. *International Journal of Serious Games*, 8(2), 67–86.
- (29) Lee M, Bahmani F, Kwan I, LaFerte J, Charters P, Horvath A, Luor F, Cao J, Law C, Beswetherick M, Long S, Burnett M, Ko A. (2014). Principles of a Debugging-First Puzzle Game for Computing Education. *Symposium on Visual Languages and Human-Centric Computing. IEEE*, 57–64.
- (30) Hyunhoon J, Hee K, Seongeun S, Jinjoong K, Changhoon O. (2019). TurtleTalk: An Educational Programming Game for Children with Voice User Interface. *ACM*, 1–6.

- (31) Kao D, Ratan R, Mousas C, Magana A. (2021). The Effects of a Self-Similar Avatar Voice in Educational Games. *Proceedings of the ACM on Human-Computer Interaction*, 5, Article 238.
- (32) Miljanovic A, Bradbury J. (2018). *A Review of Serious Games for Programming*. Serious Games, Oshawa: Springer International Publishing, 204–216.
- (33) Chase C, Malkiewich J, Lee A, Slater S, Choi A, Xing C. (2021). Can typical game features have unintended consequences? A study of players' learning and reactions to challenge and failure in an educational programming game. *British Journal of Educational Technology*, 52(1), 57–74. Saatavilla: <https://bera-journals.onlinelibrary.wiley.com/doi/epdf/10.1111/bjet.13021>. Noudettu 18.11.2021.
- (34) Mahwish S, Wajid A, Haq K, Saleem I, Shujja A. (2019). A Review of Gamification for Learning Programming Fundamental. 2019 International Conference on Innovative Computing (ICIC), 1–8. Saatavilla: https://www.researchgate.net/publication/338793871_A_Review_of_Gamification_for_Learning_Programming_Fundamental. Noudettu 18.11.2021.
- (35) Becker T. (2010). The Character of Successful Trainings with Serious Games. *International journal of emerging technologies in learning*, 5 (SI3), 20–21. Kassel University Press.
- (36) Silva J, Silveira I. (2020). A Systematic Review on Open Educational Games for Programming Learning and Teaching. *International journal of emerging technologies in learning*, 15(9), 156–172.
- (37) Valve Corporation. (2007). Portal. Saatavilla: <https://store.steampowered.com/app/400/Portal/?l=finnish>. Noudettu 18.11.2021.
- (38) Thekla, Inc. (2016). The Witness. Saatavilla: https://store.steampowered.com/app/210970/The_Witness/. Noudettu 18.11.2021.
- (39) Arvi Teikari. (2019). Baba Is You. Saatavilla: https://store.steampowered.com/app/736260/Baba_Is_You/. Noudettu 18.11.2021.
- (40) Drasute V, Dzindzeletaite G, Kelpsaite N, Drasutis S, Canbulut C. (2018). Videogames in Education: Analysis of Videogames and Apps. *Advanced Learning Technologies and Applications. Games for Education - ALTA18'*. Kaunas University of Technology, 25–28.

- (41) Brown E, Cairns P. A Grounded Investigation of Game Immersion. Extended Abstracts on Human Factors in Computing Systems. 2004. 1297–1300.
- (42) Taanila A. Määrällisen datan kerääminen. 2020. 30. Saatavilla: <http://myy.haaga-helia.fi/~taaak/t/suunnittelu.pdf>. Noudettu 18.11.2021.
- (43) Hirsjärvi S, Remes P, Sajavaara P. (2007). Tutki ja kirjoita. Kustannusosakeyhtiö Tammi. Helsinki.
- (44) Tutkimuseettinen neuvottelukunta (2019). Ihmiseen kohdistuvan tutkimuksen eettiset periaatteet ja ihmistieteiden eettinen ennakoarvointi Suomessa. Tutkimuseettisen neuvottelukunnan ohje 2019. Tutkimuseettisen neuvottelukunnan julkaisu 3/2019. Helsinki. Saatavilla: https://tenk.fi/sites/tenk.fi/files/Ihmistieteiden_eettisen_ennakoarvioinnin_ohje_2019.pdf. Noudettu 18.11.2021.
- (45) Andersen H, Mayerl J. (2019). Responding to Socially Desirable and Undesirable Topics: Different Types of Response Behaviour? Methods, data, analyses: a journal for quantitative methods and survey methodology (mda), 13(1), 7–35.
- (46) Turunen A. The Winning Teams Of Bit1 2021. Saatavilla: <https://www.bit1.fi/2021/05/25/the-winning-teams-of-bit1-2021/>. Noudettu 18.11.2021.

Liitteet

Ohjelmoinnin opetuspelin testaus

*Pakollinen



Lataa pelitiedosto osoitteesta:

<https://script-deprived.itch.io/dreamscript>

Pura tiedosto latauksen jälkeen.

Pelaat pian ohjelmoinnin opetuspeliä!

Tutkimuksella on kaksi tarkoitusta:

1. Pelitestaus eli testata pelin toimivuus
2. Selvittää toimiiko peli opetuskäytössä

Kyselyyn vastataan nimettömänä ja vastaukset käsitellään luottamuksellisesti. Tuloksia esitetään ryhmätasolla niin, ettei yksittäisiä vastauksia pystytä tunnistamaan. Ainoa henkilötieto, joka kyselyssä kerätään, on vastaajan sukupuoli.

Kyselyyn vastaaminen on vapaaehtoista ja voit myös keskeyttää tutkimuksen halutessasi.

Kiitos kun otat osaa
tutkimukseen!

Seuraa kyselyn ohjeita, ja muista vastata rehellisesti.

Tutkimus koostuu kolmesta osasta:

1. Vastaa ensin taustakysymyksiin
2. Käynnistä DreamScript sovellus ja pelaa opetuspelejä, kunnes testaukselle varattu aika umpeutuu
3. Pelitestauksen jälkeen vastaa kyselyn jälkimmäiseen osioon

ÄLÄ SULJE KYSELYÄ / SELAINTA ENNEN KUIN OLET SUORITTANUT
KAIKKI OSIOT!

Täytä tämä osio ennen pelitestausta

Tässä osiossa kerätään taustatietoja sinusta.

1. Sukupuoli

Merkitse vain yksi soikio.

- ☐ Mies
- ☐ Nainen
- ☐ Muu

2. Pelaan videopelejä vapaa-ajallani *

Merkitse vain yksi soikio.

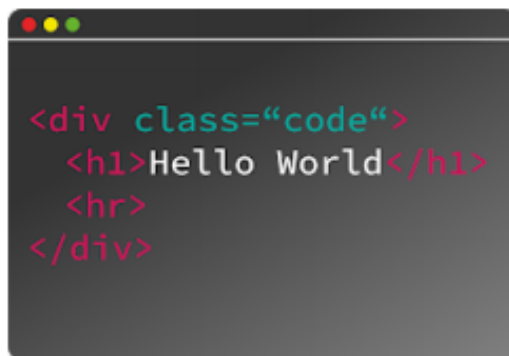
- ☐ En lainkaan
- ☐ Kerran kuukaudessa tai harvemmin
- ☐ Muutaman kerran kuukaudessa
- ☐ Muutaman kerran viikossa
- ☐ Päivittäin

3. Olen pelannut ohjelmoinnin opetuspelejä *

Merkitse vain yksi soikio.

- ☐ En lainkaan
- ☐ Alle tunnin
- ☐ Yli tunnin
- ☐ Yli viisi tuntia
- ☐ Yli kymmenen tuntia

Ohjelmakoodia voidaan kirjoittaa eri ohjelmointikielillä



```
<div class="code">
  <h1>Hello World</h1>
  <hr>
</div>
```



```
if (IsHuman)
{
    while (input.Length == 0)
    {
        Console.WriteLine("Mikä on nimesi? ");
        input = Console.ReadLine();
    }
}
else
{
    input = "Tietokone";
}
```

4. Olen kirjoittanut ohjelmakoodia *

Merkitse vain yksi soikio.

- ☐ En lainkaan
- ☐ Olen kokeillut ohjelmointia
- ☐ Olen ohjelmoinut yksinkertaisen sovelluksen
- ☐ Olen ohjelmoinut käytännöllisen / monimutkaisen sovelluksen
- ☐ Ohjelmoin säännöllisesti / harrastan ohjelmointia

5. Kuinka sopivana ammattina / harrastuksena pidät ohjelmointia itsellesi? *

Merkitse vain yksi soikio.

En lainkaan sopivana ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Erittäin sopivana

6. Kuinka vaikeana pidät ohjelmointia? *

Merkitse vain yksi soikio.

En lainkaan vaikeana ☐ ☐ ☐ ☐ ☐ Erittäin vaikeana

7. Kuinka hauskana pidät ohjelmointia? *

Merkitse vain yksi soikio.

En lainkaan hauskana ☐ ☐ ☐ ☐ ☐ Erittäin hauskana

8. I. Mielikuvani ohjelmoinnista on *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Negatiivinen	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Positiivinen

Nyt voit aloittaa pelitestauksen. Älä sulje kyselyä / selainta!

Käynnistä DreamScript sovellus.
Kun pelitestaukseen varattu aika on päättynyt, voit siirtyä seuraavaan osioon.

ÄLÄ SULJE KYSELYÄ / SELAINIA ENNEN KUIN OLET SUORITTANUT KAIKKI OSIOT!

Täytä tämä osio pelitestauksen jälkeen

Tässä osiossa kerätään tietoja pelikokemuksestasi.

9. Miten hauskaa sinulla oli pelin parissa? *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Ei lainkaan hauskaa	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Erittäin hauskaa

10. Mitkä asiat tekivät pelistä hauskan / tylsän?

11. Miten virheettömästi / bugittomasti peli toimi? *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Peli oli täysin pelikelvoton	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Peli toimi täysin virheettömästi

12. Millaisia virheitä / bugeja kohtasit pelissä?

13. Miten turhauttavana koit pelin? *

Merkitse vain yksi soikio.

	1	2	3	4	5	
En lainkaan turhauttavana	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Erittäin turhauttavana

14. Mitkä asiat turhauttivat pelissä?

15. Määrittele pelin vaikeusaste *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Liian helppo	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Liian vaikea

16. Mitkä asiat tekivät pelistä helpon / vaikean?

17. Peli lisäsi ymmärrystäni ohjelmoinnista *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Täysin eri mieltä	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Täysin samaa mieltä

18. Mitä ohjelmointiin liittyviä asioita opit pelistä?

19. Haluaisin jatkaa pelin pelaamista myöhemmin *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Täysin eri mieltä	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Täysin samaa mieltä

20. Ohjelmointiharrastus tai -ammatti on minulle sopiva *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Täysin eri mieltä	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Täysin samaa mieltä

21. Ohjelmointi on hauskaa *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Täysin eri mieltä	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Täysin samaa mieltä

22. Ohjelmointi on vaikeaa *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Täysin eri mieltä	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Täysin samaa mieltä

23. II. Mielikuvani ohjelmoinnista on *

Merkitse vain yksi soikio.

	1	2	3	4	5	
Negatiivinen	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Positiivinen

24. Miten peli vaikutti mielikuviisi ohjelmoinnista?

Ennen kuin lähetät vastauksesi, voit halutessasi jättää palautetta kyselystä

25. Anna palautetta kyselystä

Kiitos kun osallistuit tutkimukseen!

Nyt voit sulkea selaimen.
