WWW.JAMK.FI

HAVAINNOINTI OHJELMOINNIN OPPIMISEN ARVIOINTIMENETELMÄNÄ

Piia Räsänen

Kehittämishankeraportti
Huhtikuu 2010

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Ammatillinen opettajakorkeakoulu

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

KUVAILULEHTI/DESCRIPTION

Tekijä(t)	Julkaisun laji	Päivämäärä
RÄSÄNEN, Piia	Kehittämishankeraportti	25.3.2010
	Sivumäärä 42	Julkaisun kieli Suomi
	Luottamuksellisuus ( ) saakka	Verkkojulkaisulupa myönnetty (X)

Työn nimi
HAVAINNOINTI OHJELMOINNIN OPPIMISEN ARVIOINTIMENETELMÄNÄ

Koulutusohjelma
Ammatillinen opettajankoulutus

Työn ohjaaja
WEISSMANN, Kirsti

Toimeksiantaja

Tiivistelmä
Arviointi kuuluu yleensä kiinteänä osana opiskeluun. Kaikkien osapuolien kannalta on tärkeää, että arviointi on objektiivista ja antaa realistisen kuvan arvioitavan osaamisesta. Yliopistoissa käytetään arvioinnin perustana hyvin yleisesti tenttejä vaikka ne helposti antavat yksipuolisen ja pinnallisen kuvan opiskelijan tiedoista. Ohjelmointi on pohjimmiltaan ongelmanratkaisua tiettyjä lainalaisuuksia noudattaen ja hyviin lopputuloksiin päädytään hyvin erilaisia teitä. Tämän vuoksi perinteinen tenttiminen vaikuttaa erityisen huonolta arviointimenetelmältä ohjelmoinnin oppimisessa. Tässä työssä tarkastellaan ohjelmoinnin alkuopetusta ja oppimisen arviointia yliopistotasolla. Arvioinnin osalta keskitytään erityisesti opettajan ohjelmointikurssin mittaan tekemiin havaintoihin arvioinnin pohjana. Työ sisältää myös kokeilun, jossa kaksi opettajaa pyrki muodostamaan opettamiensa opiskelijoiden arvosanat puhtaasti omien havaintojensa pohjalta.

Avainsanat
arviointi, havainnointi, ohjelmointi, yliopisto-opetus

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

KUVAILULEHTI/DESCRIPTION

Author(s)	Type of publication	Date
RÄSÄNEN, Piia	Development project report	25.3.2010
	Pages 42	Language Finnish
	Confidential ( ) Until	Permission for web publication (X)

Title
OBSERVATION AS A METHOD OF ASSESSMENT IN PROGRAMMING STUDIES

Degree Programme
Vocational Teacher Education

Tutor(s)
WEISSMANN, Kirsti

Assigned by

Abstract
Assessment is usually closely tied to studying. It's important to all parties that assessment is objective and gives a truthful impression about students knowledge and skills. Examinations are very often used as a base of evaluation in universities even if it gives a shallow impression about students true knowledge. Fundamentally programming is problem-solving using certain tools and good results can be reached different ways. Because of that traditional examinations seems to be especially poor base for evaluation. In this project concentrates on first level programming studies and assessment in university. Especially teachers ability to make an assessment based on observations he/she has made about a student during a course is under observation. This assignment includes also a observation experimentation where two teachers tried to give assessments based on their observations.

Keywords
assessment, observation, programming, university teaching

Miscellaneous

SISÄLTÖ

1 JOHDANTO **2**

2 OHJELMOINTI OPPIAINEENA **4**

2.1 Ohjelmointi **4**

2.1.1 Algoritmi 5

2.1.2 Ohjelman kirjoittaminen 9

2.2 Ohjelmoinnin opiskelu **11**

2.3 Ohjelmoinnin oppimisen arviointi **13**

3 ARVIOINTI JA OPPIMINEN **16**

3.1 Arviointitavat **16**

3.2 Oppimiskäsitykset **18**

3.3 Opettajan etiikka ja arviointi **19**

4 HAVAINNOINTI OHJELMOINNIN OPISKELUSSA **21**

4.1 Temperamentti ja opiskelu **22**

4.1.1 Temperamentti ja oppiminen 23

4.1.2 Temperamentin vaikutus arvioinnissa 25

4.2 Havainnoinnin toteutus **26**

4.3 Arvioinnin muodostaminen **29**

5 HAVAINNOINTIKOKEILU **31**

5.1 Opintojakso **31**

5.2 Osallistujat **34**

5.3 Opettajien antamat arvosanat **35**

5.4 Johtopäätökset **38**

6 POHDINTA **40**

LÄHTEET **42**

1 JOHDANTO

Opiskelijoiden arviointi on yksi tärkeä osa opettajan työtä. On kaikkien edun mukaista, että annettu arvosana vastaa opiskelijan todellista osaamista mitattavasta asiasta. Mittaaminen on kuitenkin kohtuullisen vaikeaa ja erityisesti käytännön järjestelyjen takia joudutaan varsinkin yliopistotasolla käyttämään hyvin yksipuolisesti ja rajallisesti osaamista mittaavia tenttejä riippumatta siitä, kuinka hyvin ne kyseiseen tilanteeseen sopivat.

Olen opettanut useita vuosia ohjelmoinnin perusteita Kuopion yliopiston tietojenkäsittelytieteen laitoksella ja ohjannut paljon harjoituksia, joissa opiskelijat ohjelmoivat itse opettajan ohjatessa tarvittaessa. Tenttituloksia tarkastellessani sain yhä uudelleen ja uudelleen huomata, etteivät tenttitulokset ohjaamieni opiskelijoiden kohdalta juurikaan yllättäneet. Päinvastoin, huomasin pystyväni arvaamaan tenttiarvosanat valtaosan opiskelijoita kohdalta melko tarkkaan, vaikka en ollut tätä asiaa erityisesti opiskelijoita ohjatessani miettinytkään. Mieleeni tuli ajatus siitä, voisiko opettajan tekemää jatkuvaa havainnointia käyttää arviointiperusteena. Lisäksi halusin tehdä jonkinlaisen pienen dokumentoidun kokeilun siitä, kuinka hyvin opettajat todellisuudessa osaavat havaintojensa perusteella arvioida opiskelijoiden osaamista suhteessa tenttimällä saatuihin arvosanoihin.

Tässä työssä tutustutaan ensin ohjelmointiin oppiaineena sekä arviointiin ensin yleisesti ja sitten sellaisiin seikkoihin, jotka nousevat opettajan tekemän havainnoinnin osalta huomattavasti muuta arviointia enemmän esille. Erityiseen asemaan tällaisessa arvioinnissa nousee opettajan kyky erottaa opiskelijan oppiminen ja osaaminen hänen persoonastaan. Koska arviointi pohjautuu huomattavan paljon opettajan tekemiin, ei aina jälkeen päin todistettavissa oleviin havaintoihin, opettajan ammattietiikan merkitys nousee myös esimerkiksi tenttiarviointia tärkeämpään asemaan.

Tämän jälkeen esitellään suoritettu arviointikokeilu ja sen tulokset johtopäätöksineen. Viimeisessä luvussa pohditaan havainnoinnin käytön soveltuvuutta yliopiston ohjelmoinnin peruskursseille sekä yleisesti että arviointikokeilun pohjalta.

2 OHJELMOINTI OPPIAINEENA

Ohjelmointi voi sanana tarkoittaa hyvin montaa eri asiaa. Periaatetasolla ohjelmoinnilla on aina sama päämäärä: saada kone suorittamaan tiettyjä tehtäviä joko itsenäisesti tai yhteistyössä sitä käyttävän ihmisen kanssa. Teknisesti ohjelmointia on hyvin monenlaista. Puhuttaessa tietokoneiden ohjelmoinnista merkityskirjo kapenee hieman, mutta silti sana voi pitää sisällään hyvin erilaisia asioita. Tässä yhteydessä tarkastelu rajataan proseduraaliseen ja olio-ohjelmointiin. Lisäksi asioita tarkastellaan enimmäkseen Java-kielen näkökulmasta.

2.1 Ohjelmointi

Tietokone itsessään perustuu yksinkertaiseen asiaan, totuusarvoon. Tietokone siis ymmärtää vain yhden asian ja se asia on joko totta tai epätotta (true tai false). Ohjelmoinnin yhteydessä puhutaan totuusarvosta bittinä ja sitä kuvataan lukuarvona 0 tai 1 (0 = epätosi, 1 = tosi). Ketjuttamalla näitä totuusarvoja yhteen ja muuttelemalla yksittäisten bittien arvoja saadaan aikaiseksi kaikki tietokoneen toiminta. Esimerkiksi käyttäjän tallentaessa tekstinkäsittelyohjelmalla kirjoittamaansa tekstiä hän luo *tiedoston*, joka on aina pitkä ketju bittejä. Näin ollen myös kaikki tietokoneohjelmat ovat bittiketjuja, joiden bittien tila ja niiden liikkuminen tietokoneen rekistereissä saa aikaan varsinaisen ohjelman toiminnan.

Tietokoneen ohjelmointi suoritetaan pääasiassa käyttäen ns. *korkean tason ohjelmointikieliä*. Näiden kielten esitysmuoto on varsin kaukana tietokoneen sisäisestä tavasta esittää tietoa, mutta niiden tarkoitus on helpottaa ohjelmointia mm. olemalla ihmiselle mahdollisimman helposti käsitettäviä. Korkean tason kielillä kirjoitetut ohjelmat täytyy aina *kääntää* tai *tulkata* koneen ymmärtämään esitysmuotoon. Vasta tämän jälkeen ohjelmia voidaan varsinaisesti suorittaa.

Ohjelmointikieliä on olemassa suuri joukko. Osa näistä on tarkoitettu hyvin spesifisiin tarkoituksiin, mutta myös ns. yleiskieliä on useita. Kussakin ohjelmointikielessä on oma kielioppinsa eli *syntaksi*. Syntaksi sisältää käskyjoukon, joka on käytettävissä ohjelmoinnissa. Ohjelmien on aina noudatettava täydellisesti kyseisen kielen syntak-

sia. Muussa tapauksessa ohjelman kääntäminen ei ole mahdollista eikä näin saada aikaan myöskään suoritettavissa olevaa ohjelmaa. Ohjelmointikielten kielioppi on luonnollisiin kieliin verrattuna varsin yksinkertainen. Sääntöjoukko on olennaisesti pienempi eikä sääntöihin sisälly luonnollisille kielille tyypillisiä poikkeustapauksia.

2.1.1 Algoritmi

Ohjelmoinnissa varsin keskeinen käsite on *algoritmi*. Algoritmi tarkoittaa jonkin tehtävän suorittamiseksi tarvittavien toimenpiteiden kuvausta. Algoritmille annetaan korkealla tasolla kolme perusvaatimusta:

Yleisyys: algoritmin on sovelluttava määritellyn tehtävän kaikkiin tapauksiin.

Determinististyys: tehtävän ratkaisun on oltava yksikäsitteisesti määritelty ja joka vaiheessa on tiedettävä täsmälleen mitä seuraavaksi tehdään.

Tuloksellisuus: algoritmin antama tulos on aina oikea ja algoritmin suoritus päättyy aina.

Algoritmin suoritus etenee siis koko ajan kohti lopputulosta. Yleisesti algoritmin voidaan ajatella jakautuvan käskyihin ja näiden suoritusjärjestystä ohjaileviin ohjausrakenteisiin. Käskyjen suoritusjärjestystä ohjaa kolme algoritmiikan perusohjausrakennetta:

- Peräkkäisyys
- Valinta
- Toisto

Näillä kolmella välineellä voidaan kirjoittaa algoritmi käytännöllisesti katsoen mihin tahansa tehtävään. Nämä rakenteet löytyvät myös yleensä kaikista proseduraalisista ohjelmointikielistä. Ohjelmointikielet tarjoavat usein myös näistä poikkeavia ohjausrakenteita ohjelmoijan käyttöön. Tästä huolimatta ohjelmat voitaisiin kirjoittaa myös vain näihin periaatteisiin nojaten.

Peräkkäisyys algoritmiikassa tarkoittaa sitä, että käskyt suoritetaan täsmälleen niille kirjatussa järjestyksessä. Tästä järjestyksestä ei poiketa missään yhteydessä. Alla yksinkertainen algoritmi, joka kuvaa henkilön pukeutumista.

```
Ota vaatteet kaapista
Pue alusvaatteet
Laita sukat jalkaan
Pue paita
Pue housut
```

Tätä algoritmia noudattaen pukeutuminen tapahtuu joka kerta täsmälleen samassa järjestyksessä. On hyvin järkeenkäypää pukea alusvaatteet ennen paitaa ja housuja. Sukat kuitenkin voitaisi pukea myös missä tahansa muussa vaiheessa. Samoin paidan ja housujen pukeminen voitaisi tehdä toisessa järjestyksessä. Silloin ei kuitenkaan noudatettaisi tätä nimenomaista algoritmia, sillä algoritmin käskyjen suoritusjärjestys ei muutu ikinä. Samasta tehtävästä voitaisiin kuitenkin kirjoittaa hyvinkin erilainen algoritmi juuri pukeutumisjärjestystä muuttamalla. Samaan tehtävään voidaan siis kirjoittaa useammanlaisia algoritmeja, jotka kaikki toteuttavat algoritmin perusvaatimukset yhtä hyvin.

Hiukankaan monimutkaisemmissa tehtävissä pelkän peräkkäisyyden noudattaminen ei tuota riittävän tarkkaa ja yleistä ratkaisua. Ottamalla mukaan toinen perusrakenne, toisto, saadaan algoritmiin lisää vaihtelun mahdollisuutta ja sitä myöten mahdollistetaan saman algoritmin käyttö useammissa tilanteissa. Alla olevalla algoritmilla kuvataan pukeutumista ja ulos siirtymistä.

```
IF Ulkona on lämmin THEN
        Pue kesämekko
ELSE
        Pue lämpimät vaatteet
ENDIF
Mene eteiseen
IF Ulkona sataa THEN
        Ota sateenvarjo
ENDIF
Mene ulos
```

Heti alussa algoritmin suoritus haarautuu ulkolämpötilan mukaan. Jos väite "Ulkona on lämmin" on totta, suoritetaan osa "Pue kesämekko". Jos taas väite on epätosi,

päädytään pukemaan lämpimät vaatteet. Nämä ovat siis vaihtoehtoiset toimintamallit, vain toinen niistä tehdään ja päätös pohjautuu annettuun väitteeseen, joka on aina joko tosi tai epätosi. Pukeutumisen jälkeen suoritetaan aina käsky "Mene eteiseen". Seuraavaan if-lauseeseen ei sisälly else-osaa. Näin ollen sateenvarjo otetaan mukaan, jos ulkona sataa. Jos väite "Ulkona sataa" on epätosi, ei tehdä mitään vaihtoehtoista toimintaa. Lopuksi mennään joka tapauksessa ulos.

Valinnaisuuden lisäksi kompaktien algoritmien tuottamisessa toisto on olennaisessa roolissa. Algoritmeissa toistetaan usein samaa tai hyvin samankaltaista tehtävää monta kertaa. Algoritmin kirjoittaminen olisi työlästä ja seuraaminen vaikeaa, jos samaa tehtävää kirjoitettaisiin peräkkäin lukuisia kertoja. Sen sijaan yksinkertaisella toistorakenteella asia saadaan esitettyä lyhyesti ja tarkasti. Esimerkiksi kymmenen kahvikupillisen keittäminen kahvinkeittimellä:

```
Kaada vesi keittimeen
Aseta suodatinpaperi paikalleen
REPEAT 10 TIMES
        Laita mitallinen kahvia suodattimeen
ENDREPEAT
Käynnistä keitin
```

Toisinaan toistettavien kertojen määrää ei edes tiedetä etukäteen ja toiston päättyminen on kiinni algoritmin etenemisestä. Esimerkkinä sankon täyttäminen mullalla:

```
WHILE Sanko ei ole täynnä DO
        Ota multaa lapioon
        Kaada multa sankoon
ENDWILE
```

Tässä tarkistetaan ehdon "Sanko ei ole täynnä" paikkansapitävyys joka kerta ennen kuin multaa otetaan lapioon. Kun sanko täyttyy, ehto muuttuu epätodeksi toistaminen lopetetaan.

Suosituissa ns. proseduraalisissa kielissä ovat keskeisessä roolissa ohjausrakenteiden lisäksi *aliohjelmat*. Aliohjelma (proseduuri, funktio tai metodi yhteydestä riippuen) suorittaa jonkin yksikäsitteisen osatehtävän. Sen sijaan että kirjoitettaisi pitkä algoritmi, joka suorittaa jonkin tehtävän, jaetaan tehtävä osatehtäviin, joita voidaan edelleen jatkaa osatehtäviin niin kauan, että osatehtävä on hyvin spesifinen. Aliohjelman

toimintaa voidaan ohjailla *parametreilla*, joiden avulla saadaan yhdestä aliohjelmasta monikäyttöisempi. Esimerkkinä vaikkapa hyvin spesifinen kaakaojauheen annostelu kaakokuppiin:

```
MODULE Kaakaota kuppiin (lusikallisia)
        REPEAT lusikallisia TIMES
                Ota kaakojauhetta lusikkaan
                Kaada kaako kuppiin
                Sekoita
        ENDREPEAT
ENDMODULE
```

Nyt kyseistä aliohjelmaa voidaan *kutsua* eri parametrin arvoilla. Jos Liisa haluaa mietoa kaakaota, hän laittaa jauhetta vain yhden lusikallisen. Tällöin kutsu on tällainen:

```
Kaakaota kuppiin(1)
```

Jos taas Maija tahtoo vahvaa kaakota, laittaa hän kaakota kolme lusikallista kutsuen aliohjelmaa seuraavalla tavalla:

```
Kaakaota kuppiin(3)
```

Vaihtelemalla aliohjelmien parametreja saadaan sama koodi suorittamaan samankaltaisia tehtäviä. Tämä helpottaa algoritmien kirjoittamista kahdella tapaa. Ensiksikin kokonaisalgoritmista tulee lyhyempi ja kompaktimpi. Toisekseen aliohjelmilla voidaan jakaa ohjelma selkeisiin yksiköihin, joista kullakin on oma tarkoin määritelty tehtävänsä.

Aliohjelmia käytettäessä algoritmien suunnittelu etenee usein ns. *hajoita ja hallitse -menetelmän* mukaisesti. Varsinainen toteutettava tehtävä jaetaan osatehtäviksi jotka edelleen jaetaan osatehtäviksi. Tätä jatketaan niin kauan, että saadaan sopivan tarkalla tasolla olevia osatehtäviä, jotka siten toteutetaan aliohjelmina.

Näillä välineillä voidaan kuvata lähestulkoon mikä tahansa reaalimaailman ongelman ratkaisu. Reaalimaailman ongelmat, pienetkin sellaiset, ovat kuitenkin usein melko monisyisiä ja siksi täydellisiä algoritmeja on vaikea esittää. Edellä esitettyjen algoritmien kohdalla on monia virhemahdollisuuksia. Mitä jos vaatteidenpukemisalgoritmia suoritettaessa huomataan, ettei kaapissa olekaan vaatteita? Tai jos sukat eivät mahdukaan jalkaan? Mitä kahvinkeittoalgoritmin edetessä käykin niin, ettei kahvin puruja enää olekaan? Nämä ja lukuisat muut epäselväksi jääneet asiat horjuttavat algoritmi-

en tuloksellisuutta. Kaikkien näiden algoritmien voidaankin todeta pätevän vain tietyin reunaehdoin, joita ei ole tässä yhteydessä määritelty.

2.1.2 Ohjelman kirjoittaminen

Proseduraalinen ohjelmointi on imperatiivinen ohjelmointiparadigma. Sen mukaan ohjelma jaetaan itsenäisiin osiin, aliohjelmiin, joita voidaan kutsua ohjelman eri osista. Varsinainen ohjelmoijan kirjoittama lähdekoodi koostuu ohjelmointikielellä kirjoitetusta algoritmista. Kullakin ohjelmointikielellä on oma kielioppinsa eli syntaksinsa. Se antaa säännöt siihen, kuinka esimerkiksi edellisessä luvussa esitetyt keskeiset ohjausrakenteet ja aliohjelmien toteutus tehdään. Kielissä voi olla myös muunlaisia ohjausrakenteita. Kaikkien ohjelmien täytyy noudattaa syntaksia pilkuntarkasti, minkäänlaisia poikkeuksia sääntöihin ei sallita.

Varsinainen ohjelma on itsessään algoritmi, joka on toteutettu ohjelmointikielellä. Ohjelmoinnissa keskeistä on paitsi käytettävän kielen syntaksin ja erityispiirteiden hallinta, myös ongelmanratkaisutaito. Ennen kuin ongelmasta voidaan muodostaa toimiva tietokoneohjelma, täytyy keksiä ongelman ratkaisuperiaate ja pystyä kuvaamaan se yksikäsitteisellä algoritmilla. Vasta lopuksi algoritmi kirjoitetaan ohjelmointikielellä. Proseduraalinen algoritmi ei sinällään ole välttämättä sidonnainen mihinkään kieleen. Kaikki kielet nojaavat samoihin ohjausrakenteisiin, joiden avulla käytännössä kaikki yleisesti ottaen ratkaistavissa olevat ongelmat voidaan ratkoa. Algoritmin käytännön toteutuksessa nousevat sitten kielen erityispiirteet suurempaan merkitykseen. Toki taitava suunnittelija huomioi käytettävän kielen myös ongelman ratkaisuperiaatetta suunnitellessaan.

Teollisuudessa tuotettavat ohjelmat ovat suuria kokonaisuuksia, jotka yleensä sekä suunnitellaan että toteutetaan osissa. On myös tavallista, että suunnittelu hoidetaan ainakin jollain tasolla eri henkilöiden toimesta kuin varsinainen ohjelmointi. Tällä ei kuitenkaan ole merkitystä ohjelmoinnin alkeita opeteltaessa. Alussa juuri ongelmanratkaisu ja algoritmin laatiminen nousevat todella suureen rooliin. Ohjelmointikielten sääntöjen joukko on verrattaen pieni ja ulkoa opeteltavissa. Keskeisin asia on näiden sääntöjen soveltaminen siten, että annettu tehtävä saadaan suoritettua itse tehdyllä

ohjelmalla. Varsinaisen ohjelmoinnin voidaan siis sanoa olevan ongelmanratkaisua ja soveltamista.

Kuten hyvän algoritmin, myös kokonaisen ohjelman tulee täyttää tuloksellisuuden vaatimus. Ohjelman on tuotettavat oikea tulos aina. Tämän vuoksi ohjelmoijan on osattava ottaa huomioon myös kaikenlaiset epätyypilliset tilanteet sekä myös mahdolliset käytössä tapahtuvat virheet. Aivan yhtä tärkeää kuin se, että ohjelma toimii oikein sen varsinaisessa käyttötarkoituksessa on se, että se toimii hallitusti ja järkevästi myös virheellisessä käytössä. Ohjelma ei siis saa tuottaa vääriä tuloksia tai keskeyttää toimintaansa vaikka vastaan tulisikin epätyypillinen käyttötilanne.

Toinen ohjelmoinnin opetuksen alkuvaiheessa esille tuleva ohjelmointitekniikka on olio-ohjelmointi. Olio-ohjelmoinnin lähestymistapa ongelmiin on erilainen kuin proseduraalisen ohjelmoinnin. Siinä missä proseduraalisessa ohjelmoinnin algoritmi muodostaa ikään kuin reseptin, jonka mukaan ongelma ratkaistaan, olio-ohjelmoinnissa ongelmaan liittyvät toimijat ja käsitteet mallinnetaan kokonaisuuksina, olioina. Ohjelman toiminta on olioiden välistä kommunikointia. (Wikla 1998, 141) Olio kuvastaa siis jotain kokonaisuutta, esimerkiksi ohjelman käytön kannalta olennaista reaalimaailman esinettä. Tyypillisesti olio sisältää yhtä tällaista kohdetta kuvaavat tiedot sekä operaatiot sekä näiden tietojen että olion muun "käytöksen" hallintaan. Operaatiot ovat teknisesti aliohjelmia ja oliokielet sisältävät myös sekä edellisen luvun perusohjausrakenteet, että yleensä myös muita vastaavia rakenteita. Olioiden operaatiot ovat siis niin ikään ohjelmointikielellä toteutettuja algoritmeja.

Ohjelmointikielillä ja tietokoneen varsinaisella toiminnalla ei ole mitään tekemistä keskenään. Tietokoneen ymmärtämät käskyt ovat niin yksinkertaisia ja niitä vaaditaan pienenkin operaation tekemiseen niin suuri joukko, että ihmisen on huomattavasti vaikeampi hallita sellaisten käskyjen kirjoittamista ja suuren määrän takia myös virheiden riski kasvaa huomattavasti. Jotta varsinainen ohjelmointikielellä ohjelmoitu ohjelma voitaisiin suorittaa koneella täytyy se ensin saattaa muotoon, jota tietokone ymmärtää. Tätä varten suoritetaan käännösprosessi, jonka yhteydessä myös ohjelman syntaksi tarkistetaan. Ainoastaan syntaktisesti täydellinen ohjelma voidaan saattaa suoritettavaan muotoon ja vasta tässä vaiheessa päästään tarkistamaan, toimiiko ohjelma myös loogisesti ja semanttisesti oikein.

2.2 Ohjelmoinnin opiskelu

Lähes poikkeuksetta sekä proseduraalisten että olio-kielten opetus aloitetaan perusalgoritmiikasta. Tämä tarkoittaa kyseisen kielen ohjausrakenteiden huolellista läpikäymistä sekä syntaksin että merkityksen ja käytön osalta. Osa kielistä sisältää luvussa 2.1.1 esiteltyjen perusrakenteiden lisäksi muunkinlaista ohjelman suoritusta ohjaavia rakenteita. Näihin perehtyminen jätetään yleensä hieman myöhempään vaiheeseen. Ohjelmointi kuitenkin nojaa hyvin pitkälle näihin perusrakenteisiin ja niillä voidaan lähes poikkeuksetta korvata kielikohtaiset muut rakenteet. Niiden hallinta muodostaa perustan koko ohjelmoinnille, joten niiden hyvän hallinnan saavuttaminen on ensiarvoisen tärkeää.

Ohjelmoinnin alkuopetus etenee yleensä käsi kädessä algoritmiikan perusteiden oppimisen kanssa. Eri ohjausrakenteita opetellaan yksi kerrallaan siten, että ensin opiskelija oppii ymmärtämään rakenteen merkityksen, ohjelman suorituksen etenemisen rakenteessa sekä kyseisessä kielessä käytettävän syntaksin. Tämän jälkeen opiskelijan tulee oppia käyttämään ja soveltamaan rakennetta haluamiinsa tarkoituksiin.

Kun opiskelija hallitsee perusohjausrakenteet, hän voi ryhtyä soveltamaan niitä laajemmin. Pelkkä rakenteiden tekninen hallinta ei riitä, ohjelmoijan täytyy pystyä soveltamaan kuhunkin tilanteeseen sopivaa rakennetta sopivalla tavalla. Koko ajan mukana kulkee myös luvussa 2.1.1 esitetyt algoritmin perusvaatimukset, joista ei varsinaisissa tietokoneohjelmissa voida lipsua. Kaikenlaisten virhetilanteiden huomioiminen ja hallinta on aivan yhtä tärkeää kuin ohjelman oikeellinen toiminta "normaaleissa" tilanteissa. Alusta asti on tärkeää opetella hyvää ohjelmointitapaa eli hyvien ja tehokkaiden ratkaisualgoritmien hakemista sekä erilaisten virheiden ymmärtämistä ja hallintaa.

Algoritmisten perusvälineiden hallinnan jälkeen ohjelmoinnin opiskelu on hyvin konstruktiivista. Jotta ohjelmoinnissa päästään alkuun, täytyy ensin opetella pienimmät perusasiat kuten muuttujien käyttö sekä ohjausrakenteet. Tämän jälkeen uusi asia pohjautuu yleensä aina johonkin jo aiemmin opittuun, vähintäänkin juuri näihin aivan peruspalikoihin. Koska perusasioiden merkitys ei katoa pidemmällekään mentäessä, niiden sisäistäminen vahvistuu koko ajan myös vaativampien asioiden parissa liikut-

taessa. Näin opiskelussa tapahtuu koko ajan paitsi kertausta myös uusien ilmiöiden löytämistä jo aikaisemmin opitusta asiasta. Toisinaan vasta myöhemmin opittu asia antaa valmiudet ymmärtää täysin jo aiemmin käytettyjä välineitä.

Ohjelmoinnin opiskelu ei tapahdu niinkään kirjoja lukemalla, vaan käytännön tekemisen kautta. Kuten luvussa 2.1.1 on esitetty, algoritmien laadinnassa käytetään vain muutamia ohjausrakenteita, mistä seuraa, että ohjelmointikielissä on suhteellisen pieni sääntöjoukko, jonka opettelu ei ole ylenpalttisen haastava tehtävä. Tärkeämpää on kuitenkin ymmärtää näiden sääntöjen merkitys ja osata käyttää niitä ongelmanratkaisun välineinä. Tätä taitoa ei ole helppoa oppia pelkkiä kirjoja lukemalla. Toki kirjallisuuden esimerkeistä saa hyvän kuvan siitä, miten ohjelmointikielten välineitä käytetään, mutta opiskelija tarvitsee myös taitoa soveltaa esimerkissä esitettyjä tekniikoita omaan käyttöönsä. Yleisesti ottaen ohjelmoinnin opiskelussa tehdään paljon varsinaista ohjelmointia. Kielten kääntäjät antavat opiskelijoille suoraa palautetta siitä, onko ohjelma syntaktisesti oikein kirjoitettu. Käännettyä ohjelmaa suorittamalla voidaan myös konkreettisesti itse todeta ohjelman toiminnan oikeellisuus ja virheenkestävyys. Vaikka koneen tuottama palaute onkin hyvin mekaanista, se on kuitenkin oppimisen kannalta tärkeää. Opiskelijan on ymmärrettävä, miksi virhe tapahtui voidakseen paikallistaa ohjelmakoodista oikean kohdan ja korjata sen oikein.

Koska ohjelmointi on pohjimmiltaan ongelmanratkaisua, on ohjelmoinnin opiskelukin joko ongelmanratkaisua tai ongelmanratkaisun opiskelua. Ongelmanratkaisu itsessään on prosessi, joka sisältää neljä vaihetta: ongelman ymmärtämisen, ratkaisusuunnitelman laatimisen, ratkaisusuunnitelman toteutuksen ja prosessin tulkinnan. (Haapasalo 2001, 178) Tämän prosessin tarkastelu osoittaa ongelmanratkaisun olevan tärkein oppimisen ja inhimillisen ajattelun osatekijä. Toisaalta kuitenkin näyttäisi siltä, ettei ongelmanratkaisua edes voi opettaa toiselle. Tutkimustulokset kuitenkin osoittavat, että erilaisten heuristiikkojen oppiminen on mahdollista ja sitä voidaan tukea erilaisilla pedagogisilla ratkaisuilla. Erilaisten heuristiikkojen hallinta puolestaan edesauttaa ongelmanratkaisua, joten näin ollen ongelmanratkaisua on mahdollista sekä oppia että opettaa. Opettaminen kuitenkin on suhteellisen vaativaa, opettajan on hallittava laaja-alaisesti sekä itse ongelmanratkaisu että siihen liittyvät pedagogiset seikat. (Haapasalo 2001. 124-126)

2.3 Ohjelmoinnin oppimisen arviointi

Vaikka ohjelmoinnin opiskelu on luonteeltaan hyvin konstruktivistista, sen arviointi ei kuitenkaan yleensä noudata konstruktiivista oppimiskäsitystä (kts. luku 3.2). Varsinkin yliopistoissa arvioinnissa on hyvin yleisesti mukana aivan puhtaan behaviorismin piirteitä. On hyvin tavallista, että opiskelijoiden ahkerasta kotitehtävien tekemisestä palkitaan esimerkiksi tenttiarvosanan korotuksella tai ylimääräisillä tenttipisteillä. Muutenkin arviointi painottuu usein opettajan suorittamaan yksipuoliseen toimintaan, jossa arviointi suoritetaan tenttitulosten perusteella. Kuitenkin ohjelmoinnin opiskelun luonteen huomioiden arvioinnista voisi hyvinkin perustellusti löytyä niin kognitiivisen kuin konstruktiivisenkin oppimiskäsityksen mukaisia tekijöitä.

Yliopisto-opetuksessa arviointi on perinteisesti toteavaa sisältäen myös kontrolloivan arvioinnin piirteitä (kts. luku 3.1), eikä ohjelmoinnin opetus tee tässä poikkeusta. Kuitenkin motivoivan, kehittävän ja ohjaavan arvioinnin perusteet olisivat sellaisia, jotka soveltuisivat hyvin myös ohjelmoinnin oppimisen arviointiin. Tätä varten arviointia täytyisi kuitenkin kehittää pois tenttikeskeisestä kohti vuorovaikutteisempaa mallia.

Ohjelmoinnin osaamista voidaan arvioida esimerkiksi Bloomin taksonomian mukaisesti. Bloomin taksonomia on yhdysvaltalaisen Benjamin Bloomin vuonna 1956 esittämä jaottelu osaamisen eri tasoille. Bloomin mukaan osaaminen jakautuu kuudelle eri tasolle. Kunkin tason osaamisen hallinta pitää sisällään myös alempien tasojen osaamisen. Taulukossa 1 on esitettynä nämä tasot, niiden kuvaukset sekä merkitys ohjelmoinnin osaamisen arvioinnissa ohjelmoinnin perusteiden oppimisen yhteydessä. (Itä-Suomen virtuaaliyliopisto 2001-2004; University of Victoria; Starr, Manaris & Stalvey 2008)

Taso	Osaaminen	Osaaminen ohjelmoinnissa
1 Tieto (Knowledge)	Tietoisuus perusasioista, kyky luetella asiaan liittyviä termejä ja niiden kuvauksia ulkomuistista.	Osaa ja muistaa ulkoa ohjelmoinnin peruskäsitteet.
2 Ymmärrys (Comprehension)	Tavoittaa tiedon merkityksen ja tarkoituksen.	Pystyy selittämään ohjelmointiin liittyviä käsitteitä omin sanoin.
3 Sovellus (Application)	Osaa käyttää tietoa hyväkseen ja soveltaa uusissa tilanteissa. Löytää erilaisia keinoja tehtävästä suoriutumiseen.	Osaa soveltaa perustason osaamista, kuten eri ohjausrakenteita uusissa tilanteissa.
4 Analyysi (Analysis)	Näkee kokonaisuuksia ja yhteyksiä (myös piilotettuja) asioiden välillä. Kykenee erottamaan olennaisen asian epäolennaisista.	Osaa jäsentää tehtävän osiin siten, että ne muodostavat järkevän ja ymmärrettävän kokonaisuuden.
5 Synteesi (Synthesis)	Pystyy tuottamaan uusia ratkaisuja vanhojen pohjalta. Kykenee yhdistelemään tietoa eri aihealueilta.	Osaa rakentaa hallitsemistaan välineistä uusia kokonaisuuksia toteutustasolle asti.
6 Arviointi (Evaluation)	Osaa laatia arviointikriteerit ja käyttää niitä eri ratkaisujen vertailuissa.	Pystyy arvioimaan ratkaisujen tasoa sekä vertailemaan perustellusti ratkaisuja toisiinsa.

Taulukko1: Bloomin taksonomia ohjelmoinnin osaamisen arvioinnissa

Arvioitaessa opiskelijan suoriutumista ohjelmointikurssilla on tarkoituksenmukaista laatia oman taksonomiansa kullekin kurssin osaamistavoitteissa luetellulle kohdalle erikseen. (Starr, Manaris & Stalvey 2008) Kuvauksen tarkkuudesta riippuen jopa pienempienkin kokonaisuuksien muodostaminen voi olla perusteltua.

Ohjelmoinnin osaaminen voidaan jakaa Bloomin taksonomian mukaan kolmeen osaamistasoon. Tyydyttävä osaaminen (beginner) pitää sisällään tasojen 1 ja 2 hallinnan, hyvä osaaminen (intermediate) tasojen 3 ja 4 hallinnan ja erinomainen osaaminen (expert) tasojen 5 ja 6 hallinnan. (Starr, Manaris & Stalvey 2008)

Arvioitaessa ohjelmoinnin perusteiden oppimista taulukon 1 kuvaama ohjelmoinnin osaaminen on liian korkealla tasolla kuvattua. Ensimmäisellä ohjelmointikurssilla ei

voida vielä olettaa opiskelijan kykenevän hallitsemaan suuria kokonaisuuksia ja pystyvän analyyttisesti ja monipuolisesti vertailemaan erilaisia ratkaisuja toisiinsa. Hyvin matalalla tasolla, yksittäisen ohjausrakenteen opettelussa, taksonomia voitaisi muodostaa esimerkiksi taulukossa 2 kuvatulla tavalla. (Starr, Manaris & Stalvey 2008; Thompson ym. 2008)

Taso	**Osaaminen**
1 Tieto	Tuntee ohjausrakenteen määritelmän.
2 Ymmärrys	Osaa selittää rakenteen merkityksen.
3 Sovellus	Kykenee seuraamaan algoritmin, jossa on käytetty kyseistä rakennetta, suoritusta.
4 Analyysi	Ymmärtää rakenteen eri osat ja niiden merkityksen kokonaisuuden kannalta.
5 Synteesi	Osaa suunnitella ja toteuttaa ongelman ratkaisevan algoritmin käyttäen kyseistä rakennetta.
6 Arviointi	Pystyy arvioimaan jonkin kyseisellä rakenteella toteutetun ratkaisun hyvyyttä.

Taulukko2: Bloomin taksonomia yksittäisen ohjausrakenteen oppimisessa

Käytännössä kunkin yksittäisen osaamistavoitteen taksonomia on jotain taulukoissa 1 ja 2 esitettyjen ääripäiden väliltä. Yksi osaamistavoite voi jakautua moniin itsenäisesti arvioitaviin osiin, joista sitten muodostetaan kokonaisarvostelu. Esimerkiksi osaamistavoite ”hallitsee ohjelmoinnin perusohjausrakenteet” voitaisi jakaa erikseen eri ohjausrakenteiden hallinnan arviointiin taulukon 2 mukaisen taksonomian mukaisesti ja näiden perusteella muodostettaisiin lopullinen arviointi oppimistavoitteen täyttymiselle.

Bloomin taksonomia ei ole ainoa eikä välttämättä edes paras taksonomia ohjelmoinnin osaamisen arviointiin. Toisaalta se soveltuu tehtävään hyvin, sillä se on yksinkertainen ja perustuu selkeisiin ja tunnistettaviin kognitiivisiin seikkoihin. Toisaalta taas sen mukaan laaja-alainen ongelmanratkaisu ja ratkaisujen tehokkuuden arviointi edellyttää korkean tason osaamista, vaikka näiden pitäisi olla ohjelmoinnin peruspilarit. (Fuller ym. 2007) Näin ollen se ei ehkä sovi korkeamman ohjelmointiosaamisen arviointiin niin hyvin kuin perusteiden oppimisen arviointiin.

3 ARVIOINTI JA OPPIMINEN

Arvioinnilla on pitkät perinteet opiskelun ja oppimisen mittarina. Jo lainsäädännössä määrätään, että peruskoulussa, keskiasteella että korkea-aseteella opiskelijoiden suorituksia tulee arvioida. (Finnlex: Perusopetuslaki 22 §, Lukiolaki 17 §, Laki ammatillisesta koulutuksesta 25 §, Ammattikorkeakoululaki 27 §, Yliopistolaki 44§). Lainsäädäntöön on kirjattu myös arviointien tarkkoja muotoja (Finnlex: Perusopetusasetus 10 §, Lukioasetus 6 §, Asetus ammatillisesta koulutuksesta 10 §). Yliopisto- ja ammattikorkeakouluopintojen arviointia ei asetuksissa rajata yhtä tarkoin kuin alempien asteiden arviointia.

Perimmiltään arvioinnin voidaan ajatella olevan oppimistulosten vertailua oppimistavoitteisiin. Sinällään yksinkertainen määritelmä sisältää kuitenkin monta muuttujaa lähtien oppimiskäsityksistä, sekä tiedon ja osaamisen käsitteistä. Lisäksi tulee muistaa arvioinnin mahdollisuudet oppimisprosessia tukemassa. Arvioinnin avulla on mahdollista ohjata, kannustaa ja motivoida opiskelijaa. (Koli & Silander 2002).

3.1 Arviointitavat

Toteava arviointi on hyvin yleinen arviointitapa. Siinä yksinkertaisesti todetaan jokin asia, hyvin tyypillisesti opiskelijan osaamistaso. Hyvin tyypillisesti tenttien arviointi on toteavaa.

Motivoivassa arvioinnissa on arvioinnin lisäksi aina mukana opiskelijaa kannustava tekijä. Arvioija ja arvioitava arvioivat yhdessä keskustellen annettujen tavoitteiden toteutumista. Opettajan tuki ja kannustus sekä asetettujen tavoitteiden saavuttamisen todennäköisyys voivat toimia opiskelijaa motivoivina tekijöinä. Motivoiva arviointi vaatii onnistuakseen luottamussuhdetta, opiskelijan on voitava luottaa opettajan arvion oikeellisuuteen.

Ohjaava arviointi auttaa opiskelijaa tekemään omaa oppimistaan edistäviä ratkaisuja. Ohjaava arviointi on tyypillisesti jatkuvaa, oppimisprosessia seurailevaa toimintaa.

Arviointi tukee opiskelijan yksilöllistä opiskelua ja antaa eväitä itsenäiselle oppimisprosessin hallinnalle.

Kehittävä arviointi ajoittuu koko oppimisprosessin ajalle. Sekä opettaja että opiskelija ovat prosessissa toimijoita, jotka ovat valmiita kehittämään toimintaansa. Molemmat osallistuvat arviointimenetelmien valintaan ja arviointitulosten määrittelyyn. Lopputulos ei siis välttämättä ole kaikille sama, vaan se määräytyy ja kehittyy yksilöllisesti oppimisprosessin varrella.

Kontrolloivassa arvioinnissa arvioinnin kohteena on opiskelijan suoritus ja opettajan kiistattomana roolina suorituksen arviointi ja arvosanan antaminen. Kontrollin tehtävänä on motivoida ja ohjata opiskelijaa toimintansa suuntaamisessa. Varsinainen uhka huonon arvioinnin kielteisistä seurauksista ei ole kontrolloivan arvioinnin edun mukaista, se saattaa helposti vaikuttaa negatiivisesti opiskelijan motivaatioon. Myöskään liiallinen opiskelijan ja opettajan välinen yhteistyö ja vuorovaikutus arvioinnin yhteydessä ei tue kontrollia.

Valikoivan arvioinnin tarkoituksena on valikoida kehittämisen arvoisia asioita arvioinnin kohteena olevasta materiaalista. Arvioijan valta-asema on selkeä, hän valitsee mitä arvioidaan ja miten arviointi suoritetaan. Hyvin tyypillistä valikoivaa arviointia noudattavat esimerkiksi valintakokeet, joissa valitaan jatkoon pääsevät hakijat heidän suoriutumisensa perusteella.

Ennustavassa arvioinnissa pyritään arvioimaan olemassa olevien tietojen varassa, millaiseksi arvioitava tulee kehittymään. Ennustavuus kytkeytyy voimakkaasti ohjaustehtävään.

Vaikka erilaiset arviointitavat onkin yllä jaettu seitsemään eri tapaan, ne eivät ole toisestaan riippumattomia tai toisensa poissulkevia. Varsinaisessa käytännön tason arvioinnissa voi yhdistyä piirteitä monista eri arviointitavoista. Samaa kokonaisuutta arvioitaessa voidaan myös käyttää eri arviointitapoja eri osioiden arviointiin.

3.2 Oppimiskäsitykset

Opettajalla on opetuksen taustalla aina tietoisesti tai tiedostamatta jokin käsitys siitä, miten oppiminen tapahtuu. Tämän käsityksen mukaan muotoutuu itse opetus ja sen jatkona tuleva arviointi. Opetus ei välttämättä noudattele puhtaasti mitään oppimiskäsitystä, se voi olla näiden yhdistelmä erilaisin painotuksin. (Itä-Suomen verkkoyliopisto 2001 – 2004) Olennaista kuitenkin olisi, että arvioinnin oppimiskäsitys kulkisi käsi kädessä opetuksen kanssa.

Behavioristisen oppimiskäsityksen mukaan opettaja on hyvin keskeinen toimija oppimisprosessissa. Hän toimii aktiivisena tiedon jakajana ja tilanteiden kontrolloijana. Arviointi toimii hyvin paljon itseisarvona ja sen suorittaa yksinomaan opettaja. Opiskelijaa palkitaan hyvästä suoriutumisesta hyvällä arvioinnilla. Arviointi seuraa suoritusta mahdollisimman pian, jotta palkkio pysyisi mahdollisimman suurena ja houkuttelevana. (Verkkoluotsi) Arvioinnissa voi olla myös määrällisiä piirteitä, opiskelijoita kannustetaan tekemään tehtäviä siten, että arvosanaa nostetaan ahkeran tehtävien tekemisen seurauksena.

Kognitiivisen oppimiskäsityksen mukaan mielekkään oppimisprosessin aloittaa käytännön elämä ongelma tai ristiriita. Oppija tiedostaa, etteivät hänen tietonsa ja kykynsä riitä sen ratkomiseen ja ryhtyy hankkimaan tarvittavia tietoja ja taitoja. (JAMK) Opettaja antaa palautetta oppimisprosessin etenemisestä sen eri vaiheissa arvioiden laajasti myös opiskelijan oppimisstrategioiden hyödyntämistä ja mentaalisten mallien muodostumisen sekä ongelmanratkaisun etenemisen osalta. Varsinaiseen arviointiin osallistuu myös opiskelija ja arviointi kohdistuu kehittymiseen ja sen prosesseihin. (Verkkoluotsi)

Humanistinen oppimiskäsitys näkee oppimisen hyvin laajasti oppijaa koskettavana eri aisteja, kokemuksia ja tunteita hyödyntävänä prosessina. Uuden asian oppiminen nähdään vanhan tiedon syventämisenä ja samalla yksilön kokemuspiirin laajentamisena. Opetus on keskustelevaa ja opiskelijalla on koko ajan aktiivinen rooli. Arvioinnissa tärkeässä roolissa on kokemuksellisuus, eli miten tavoitteet koetaan saavutetuiksi. Arvioinnissa suuri paino on opiskelijan tekemillä itsearvioinneilla. Yksikäsittei-

siä kriteereitä oppimisen arviointiin ei humanistisen oppimiskäsityksen myötä ole mahdollista muodostaa. (Itä-Suomen verkkoyliopisto 2001 – 2004)

Konstruktivistinen oppimiskäsitys näkee oppimisen aktiivisena tiedon rakentamisprosessina. Uudet asiat havainnoidaan aikaisemmin omaksuttujen tietojen ja kokemusten pohjalta. Varsinainen oppiminen on seurausta oppijan omasta aktiivisesta toiminnasta ja ongelmanratkaisu on keskeisessä roolissa. Konstruktivistisen oppimiskäsityksen mukaisesti arvioinnin tulisi olla hyvin joustavaa ja monipuolista oppimisprosessin arviointia. (JAMK)

3.3 Opettajan etiikka ja arviointi

Etiikka eli moraalifilosofia voidaan ajatella systemaattiseksi kyvyksi ymmärtää moraalisia käsitteitä kuten oikea, väärä, sallittu, pitäisi, hyvä ja paha. Lisäksi se on pyrkimystä oikeaan käyttäytymiseen sekä hyveiden ja elämän arvojen tunnistamiseen. Etiikka ei siis ole mitään ulkoa opeteltavaa, vaan enemmän sisäinen ja sitouttava asia, johon liittyy jatkuvaa pohdintaa. Etiikan keskeisenä sisältönä on moraali, jonka voidaan puolestaan katsoa kumpuavan yksilön arvomaailmasta. Tämä muodostuu ihmisen sisällä perimän ja kokemusten ohjaamana ja on läsnä joka hetki ihmisen elämässä. Kaikilla ihmisillä on siis moraali, vaikka kaikkia ihmisiä ei voidakaan pitää moraalisina. Etiikka ei sen sijaan välttämättä ole kaikilla ihmisillä sillä etiikka on tietoista (Atjonen 2007, 12-15).

Etiikka, erityisesti opettajan ammattietiikka, nousee voimakkaasti esille myös puhuttaessa arvioinnista. Opettajan tehtävänä on opetuksen ja oppimisen mahdollisuuksien edistäminen yksilön parhaaksi. (OAJ) Opettajan rooli oppimiskokemuksissa on merkittävä ja voi luoda niihin hyvinkin vahvan varsinaisesta opittavasta asiasta erillään olevan positiivisen tai negatiivisen tunnelatauksen. Näin opettaja vaikuttaa tahtomattaankin jokapäiväisessä työssään arvioinnin kohteena olevaan asiaan, oppimiseen.

Arviointiin liittyvä etiikka on asiana erityisen merkityksellinen. Perinteisesti arvioijalla on ollut yksikäsitteinen valta-asema arvioitavaan nähden. Nyttemmin varsinkin peruskoulun jälkeisissä opinnoissa arvioinnissa on kuitenkin siirrytty suuntaan, jossa

arviointivastuu ja -valta siirtyy osittain myös opiskelijoille itsearvioinnin ja vertaisarvioinnin muodoissa. Kuitenkin viime kädessä arvosanan määrää opettaja, joka sinällään jo asettaa hänet tietynlaiseen valta-asemaan opiskelijaan nähden. Tämä antaa opettajalle paitsi tiettyä valtaa myös valtavaa vastuuta. Arvosanoilla voi kuitenkin olla suuri merkitys opiskelijan tulevaisuuden kannalta. Tätä valtaa opettaja ei saisi ikinä käyttää väärin ja arvioinnin oikeudenmukaisuus ja siihen liittymättömistä asioista irrottaminen onkin yksi opettajan ammattieettisyyden kulmakiviä. (Niemi, OAJ)

Arvioinnin kannalta tärkeitä arvoja ovat oikeudenmukaisuus, validius, reliaabelius ja läpinäkyvyys. Opettajan pitäisi pystyä takaamaan, että opiskelijat arvioidaan samanarvoisesti ketään suosimatta tai syrjimättä. Samoin opettajan on kyettävä suorittamaan arviointi niin, että arviointi todella kuvastaa sitä asiaa, jota haluttiin arvioida sulkien pois sattumanvaraisuuden. Hyvässä arvioinnissa ei myöskään saa tulla opiskelijoille yllättäviä vaatimuksia tai yhteyksiä, joista ei ole etukäteen ilmoitettu. Lisäksi hyvä arviointi myös motivoi opiskelijoita oppimaan. Opiskelijoille tulee antaa reilu mahdollisuus osoittaa erinomainen osaamisensa ja toisesta päästä taas sopivalla tavalla vaatia riittävää osaamista hyväksymisen edellytyksenä. (Atjonen 2007, 34-36)

Hyvän opettajan täytyy siis löytää ammattietiikkansa ja toimia sen mukaisesti voidakseen toteuttaa hyvää arviointia. Tätä varten opettajan täytyisi kyetä olemaan täysin puolueeton. Tämä on kuitenkin vaatimuksena hyvin suuri, ehkä jopa mahdoton. Kuitenkin jokainen opettajan havainto peilautuu hänen arvoihinsa ja siten arvot ovat mukana myös näiden havaintojen pohjalta tapahtuvassa arvioinnissa. Joko tietoisesti tai tiedostamatta. (Atjonen 2007, 204) Jossain määrin subjektiivisuus on arvioijallakin vain inhimillistä, mutta sitä ei saa päästää haitallisiin mittoihin.

4 HAVAINNOINTI OHJELMOINNIN OPISKELUSSA

Jo yhdeksänkymmentäluvulla Erkki Lahdes toteaa kouluissa tapahtuvan arvioinnin helposti päätyvän korostamaan liikaa kirjallisia kokeita, vaikka valtaosa formatiivisesta arvioinnista tapahtuu tunneilla opettajan seuratessa oppilaiden työskentelyä. (Kari 1994, 206). Korkeammilla asteilla kokeiden, tenttien sekä erilaisten tehtävien merkitys arvioinnissa kasvaa yleensä entisestään, yliopistoissa opintojaksojen arviointiperusteissa on todella harvoin mitään sijaa opettajan tekemillä havainnoilla ja arviointi on kaikin puolin toteavaa. Monissa paikoissa tämä on ymmärrettävääkin, opetusryhmäkoot, opetustekniikat sekä opettajan ja opiskelijan välisen kontaktin määrä ja laatu ovat yliopistossa todella erilaisia kuin peruskoulussa. Luonnollisesti opettajan arviointi opiskelijan suoriutumista havainnoimalla on mahdotonta, ellei yhteistä aikaa sopivissa olosuhteissa ole riittävästi. Jo tämä sulkee tällaisen arviointimenetelmän pois suuresta osaa yliopisto-opintojaksoja. Se ei kuitenkaan tarkoita, ettei tällaista arviointimenetelmää voitaisi käyttää missään.

Havainnointiin perustuva arviointi ohjelmoinnin opetuksessa poikkeaa arviointimenetelmänä tenttiin perustuvasta arvioinnista merkittävästi. Kuten aina, myös ohjelmoinnissa arvosana muodostuu tällä tavoin pitkän ajan kuluessa ja opettajalla on arvioinnin tueksi käytössään enemmän materiaalia. Koska opettaja seuraa opiskelijan työskentelyä useaan eri kertaan koko oppimisprosessin ajan, käsitys nimenomaan ohjelmointitaidosta ongelmanratkaisu mukaan luettuna on hyvin erilaista kuin muutaman tentityn tehtävän perusteella. Opettajalla on mahdollisuus myös selvittää opiskelijan ajatuksia, ongelmanratkaisukykyä ja ohjelman tuottamisprosesseja eri tavalla kuin tenttitilaisuudessa.

On tärkeää, että arviointi on arviointimenetelmästä riippumatta tasapuolista, avointa sekä opiskelijoiden osaamista objektiivisesti mittaavaa. Havainnointiarvioinnissa on monia seikkoja, jotka voivat vaikuttaa heikentävästi näiden vaatimusten toteutumiseen. Opettajan on tiedostettava nämä huolellisesti voidakseen eliminoida niiden vaikutuksen mahdollisimman hyvin.

Arvioinnin läpinäkyvyys ja avoimuus on selkeästi yksi jatkuvan arvioinnin ongelmakohtia. Tenttien osalta arviointi on aina helppoa perustella jälkikäteen. Sen sijaan kaikkia niitä seikkoja, joiden perusteella opettaja on havainnointiin pohjautuvan arvionsa antanut, on vaikeampi, käytännössä mahdotonta, osoittaa jälkikäteen.

Ohjelmoinnin opiskelu sisältää lähes väistämättä itsenäistä ohjelmointiharjoittelua. Näiden valmiiden, tai opiskelijan kykyjen mukaisten vajaaksi jääneiden, tuotosten tarkastelu antaa myös opettajalle hyvää tietoa opiskelijan osaamisesta. Valmiista lähdekoodista ei tietenkään näy ohjelman suunnitteluprosessi, mutta valittu ratkaisuperiaate kuitenkin kertoo opiskelijan algoritmisen ajattelun tasosta ja laadusta.

Tässä luvussa perehdytään tarkemmin havainnointiin ja siihen liittyviin ilmiöihin. Ensin tutustutaan temperamenttiin, joka on sekä opettajan että opiskelijan puolelta sekä itse havainnointiin että oppimiseen vaikuttava sisäinen tekijä. Tämän jälkeen pohditaan havainnoinnin järjestämistä käytännössä sekä arvioinnin muodostamista havainnoinnin pohjalta.

4.1 Temperamentti ja opiskelu

Temperamentti muodostuu joukosta ominaisuuksia, jotka määräävät henkilön tavan käyttäytyä ja reagoida. Jokainen ilmentää kutakin temperamenttipiirrettä itselleen ominaisella tasolla, joka on synnynnäinen ja pysyvä. Sinällään yksittäiset temperamenttipiirteet eivät niinkään sanele ihmisen lopullista temperamenttia, olennaista on piirteiden voimakkuus ja niiden yhdistelmä. Kullakin temperamenttipiirteellä on ääripäänsä ja jokaisella ihmisellä kyseisen temperamentin ilmeneminen asettuu jonnekin näiden ääripisteiden välillä olevalle janalle. Näin muodostuu henkilön *temperamenttiprofiili*. Kaikki temperamenttipiirteet noudattavat väestössä normaalijakaumaa. Näin ollen keskiarvot ovat yleisiä ja kunkin temperamenttipiirteen ääritapaukset harvinaisia. Ihmisen temperamentin katsotaan muodostuvan niistä piirteistä, joiden osalta hän poikkeaa mahdollisimman paljon keskiarvosta. Keskiarvoisesti ilmenevät temperamenttipiirteet eivät ilmennä hänen henkilökohtaista temperamenttiaan, vaikka ne hänellä olemassa olevia piirteitä ovatkin. (Keltinkangas-Järvinen 2006, 22-25)

Lapset ilmentävät temperamenttiaan usein suhteellisen vapaasti. Iän myötä ihminen oppii hallitsemaan temperamenttiaan niin, etteivät sen mukaiset reaktiot pääse kaikissa tilanteissa esille. Perustemperamentti ei kuitenkaan taustalta muutu mihinkään. Ihminen reagoi asioihin oman temperamenttinsa mukaisesti, ainoastaan näkyvä reaktio on hallitumpi. Kuitenkin nämä synnynnäiset reaktiot ovat aikuisiälläkin taustalla samanlaisina ja yhtä voimakkaina kuin lapsena ja vaikuttavat siten käytännössä kaikkeen ihmisen elämässä, myös oppimiseen ja opiskeluun.

4.1.1 Temperamentti ja oppiminen

Temperamentilla ja kognitiivisilla kyvyillä ei ole suurtakaan yhteyttä keskenään. Näin ollen temperamentin ei pitäisi suoranaisesti vaikuttaa oppimiseen millään tavalla. Temperamentti kuitenkin määrää sen, miten henkilö suhtautuu oppimiseen ja opiskeluun. Lisäksi on niin, että opetusjärjestelyt tukevat tietynlaisia temperamenttipiirteitä. Koska temperamenttipiirteet väistämättä vaikuttavat oppimistapaan, tulee niiden vaikutus esille juuri annetun opetuksen kautta. Oppimiseen ja opiskeluun vaikuttavia temperamenttipiirteitä on havaittu olevan ainakin kahdeksan: *sensitiivisyys, aktiivisuus, sopeutuvuus, sinnikkyys, häirittävyys, rytmisyys, lähestyminen/vetäytyminen* ja *intensiivisyys*. Yhteenveto näistä temperamenttipiirteistä ja niiden ääripäistä on esitettynä taulukossa 3. (Keltinkangas-Järvinen 2006, 60-62)

Temperamenttipiirre	Piirteen korkea ilmenemistaso henkilössä	Piirteen matala ilmenemistaso henkilössä
Sensitiivisyys kuvastaa henkilön taipumusta huomioida ympärillään tapahtuvia asioita. Sisältää myös emotionaalisen ja sosiaalisen herkkyyden.	Huomaa pienimmätkin ympärillä tapahtuvat muutokset ja ärsykkeet. Reagoi herkästi erilaisiin aistiärsykkeisiin. Herkkä tulkitsemaan sosiaalisia ja nonverbaaleja viestejä.	Ei reagoi herkästi ympäristön ärsykkeisiin eikä huomioi pieniä muutoksia aistimuksissa. Tulkitsee heikosti nonverbaalia viestintää ja sosiaalisia tilanteita.
Aktiivisuus kuvastaa sitä, kuinka suuren määrän energiaa henkilö käyttää kullakin hetkellä työn alla olevaan toimintaan.	Toimii, elehtii ja puhuu voimakkaasti, hiljaa aloillaan olo vaikeaa. Innostuu herkästi uusista asioista	Toiminnot hitaita ja vähäeleisiä.
Sopeutuvuus kuvastaa henkilön kykyä sopeutua yllättäviin tilanteisiin ja uusiin asioihin sekä muutoksen vastustamisen voimakkuutta.	Helppo omaksua uusia asioita ja toimintamalleja. Ei tarvitse ennakkosuunnitelmia, suuretkaan muutokset esimerkiksi aikatauluissa eivät aiheuta ongelmia.	Yllätykset tuntuvat vaikeilta hallita ja sietää. Muutos totutuissa rutiineissa saattaa lamauttaa kokonaan.
Sinnikkyys kuvastaa henkilön sitkeyttä, peräänantamattomuutta sekä tarvetta saattaa aloitetut tehtävät loppuun.	Pyrkii aina saattamaan loppuun aloittamansa tehtävän vaikka olosuhteet ja/tai muut henkilöt olisivat sitä vastaan. Vaikeaa vaihtaa tehtävästä toiseen kesken kaiken.	Lyhytjänteinen, aloitetut tehtävät jäävät helposti kesken varsinkin jos niiden tekemisessä ilmenee ongelmia.
Häirittävyys kuvastaa henkilön taipumusta keskeyttää menneillään oleva toiminta uuden ärsykkeen vuoksi.	Huono keskittymiskyky, uudet asiat syrjäyttävät menneillään olevat. Asiat unohtuvat ja jäävät kesken helposti aina uusien tulviessa mieleen.	Hyvä keskittymiskyky, ulkoiset ärsykkeet eivät pääse häiritsemään tehtävän suorittamista.
Rytmisyys kuvastaa henkilön sekä henkilön fysiologisen kellon tarkkuutta että hänen jokapäiväisen elämänsä säännönmukaisuutta.	Säännöllinen elämänrytmi, siisteys ja järjestelmällisyys. Sekä fysiologisten että arkipäiväisten toimien ennustettavuus.	Fysiologisen rytmin puuttuminen. Epäjärjestelmällisyys ja -siisteys. Toimien ennaltaarvaamattomuus.
Lähestyminen kuvastaa henkilön taipumusta olla aktiivisesti kiinnostunut uusista asioita kun taas **vetäytyminen** kuvastaa vastakohtaisesti taipumusta olla varautunut uusia asioita kohtaan.	Kiirehtii innokkaasti ja uteliaasti uusia asioita kohti. Ulospäinsuuntautunut ja sosiaalinen.	Haluaa tutustua uusiin asioihin ensin matkan päästä ja hitaasti. Uudessa seurassa ujo.
Intensiivisyys kuvastaa voimaa, jolla henkilö ilmaisee tunteitaan ja mielialojaan.	Tunneilmaisut voimakkaita, taipumus ylireagoida. Esiintymistaipumusta, jopa aggressiota.	Tunteiden ilmaiseminen vaikeaa ja kontrolloitua. Ulkopuolisten on vaikea arvioida mielialoja.

Taulukko 2: Opiskelussa keskeiset temperamenttipiirteet

4.1.2 Temperamentin vaikutus arvioinnissa

Temperamentti siis selkeästi vaikuttaa henkilön oppimiseen ja oppimistaitoihin ja näin ollen väkisinkin myös arviointiin. Jos temperamentti vaikuttaa opiskeluun haitallisesti, se ei voi olla heijastumatta myös opiskelumenestykseen ja päinvastoin. Tärkeä kysymys on kuitenkin se, vaikuttaako temperamentti itse arviointiin erityisesti, jos arvosanan antamisen perusteena on opettajan näkemys opiskelijan osaamisesta. Kuten edellisessä luvussa todetaan, temperamenttipiirteidensä ansiosta eri opiskelijat saattavat toimia opetustilanteissa hyvin eri tavoin, joten he näyttäytyvät myös opettajille hyvin erilaisina. Hyvin aktiiviset opiskelijat tarjoavat opettajalle arviointiperusteita jopa tarpeellista enemmän, kun taas hyvin matalasti aktiivisista ja vetäytyvistä opiskelijoista ei välttämättä saa kunnollista materiaalia arvioinnin perustaksi.

Tässä yhteydessä törmäämme hyvin voimakkaasti arvioijan rooliin ja merkitykseen. Tässä yhteydessä on hyvä erottaa opiskelijoista kaksi ominaisuutta, *oppijuus* ja *opetettavuus.* (Keltinkangas-Järvinen 2006, 141) Hyvä oppija on opiskelija, joka omaksuu opetettavat asiat hyvin ja nopeasti. Hyvä opetettava puolestaan on opiskelija, joka noudattaa opettajan ohjeita tarkasti ja opiskelee hänen antamansa mallin mukaisesti. Hyvä oppija ei kuitenkaan välttämättä ole hyvä opetettava eikä hyvä opetettava hyvä oppija. Eittämättä opettajan kannalta on mukavinta, jos opiskelija on sekä hyvä oppija että hyvä opetettava. Toisaalta mukaan mahtuu myös niitä, jotka eivät ole kummassakaan hyviä. Joka tapauksessa yhteys sekä oppijuuden että opetettavuuden ja temperamentin välillä on selvä. On myös hyvin oletettavaa, että opettajalle muodostuu miellyttävämpi kuva opiskelijasta, joka on hyvä oppija kuin sellaisesta, joka ei ole. Hyvän oppijan temperamenttipiirteistä puhutaan edellisessä luvussa. Hyvän opetettavan vallitsevia temperamenttipiirteitä on sen sijaan mahdotonta luetella yksikäsitteisesti, sillä tässä on väkisin opettajakohtaisia eroja. Näin ollen temperamentti vaikuttaa selvästi siihen kuvaan, joka opiskelijasta opettajalle muodostuu ja joka on pohjana myös arvioinnille. Jotta arviointi olisi onnistunutta, opettajan pitäisi pystyä tunnistamaan nämä seikat ja siten kitkeä vaikutus pois annettavista arvioinnista.

Vielä yksi merkittävä seikka opettajan antamassa arvioinnissa on opettajan oma temperamentti ja erityisesti sen yhteensopivuus opiskelijoiden temperamentin kans-

sa. Laajemmin voidaan puhua ihmisestä psykofyysisesnä kokonaisuutena, jonka osana on myös temperamentti. On selvää, että kaikkien ihmisten välillä "kemiat" eivät yksinkertaisesti kohtaa. Näin voi käydä myös opettajan ja opiskelijan välillä. Opiskelija voi olla sekä hyvä oppija että hyvä opetettava, mutta jostain syystä opettaja ei hänestä ja hänen tavastaan toimia silti pidä. Sama pätee myös toisin päin, joku opiskelija saattaa olla opettajasta todella mukava ja erityisesti jos hän on hyvä opetettava, vaikuttaa myös ahkeralta ja tunnolliselta opiskelijalta, vaikkei varsinaista oppimista juurikaan tapahtuisi. Tällaisissa tapauksissa on helppoa sortua arvioimaan opiskelijaa hänen persoonansa mukaan varsinaisen osaamisen sijaan. Myös tämä inhimillinen seikka opettajan tulee tiedostaa ja pystyä sulkemaan pois arvioinnista.

4.2 Havainnoinnin toteutus

Perustana havainnointiin perustuvalle arvioinnille voidaan pitää sitä, että opettaja ja opiskelija ovat riittävästi tekemisissä keskenään ja opettajalla on mahdollisuus seurata myös opiskelijan työskentelyä. Kuitenkaan paljonkaan yhteisen ajan jälkeen opiskelijoiden arviointi ei ole välttämättä yksinkertaista, mukaan tulee helposti varsinaiseen asiaan liittymättömiä inhimillisiä tekijöitä. Tutkimuksissa on todettu, että opettajat arvioivat tietyntyyppisiä oppilaita lähes poikkeuksetta yli näiden objektiivisesti mitatun osaamisen. Myös tietynlaisten ominaisuuksien on todettu vaikuttavan opettajan arviointiin heikentävästi. (Keltinkangas-Järvinen 2006, 144)

Käytännössä tällainen havainnointiin perustuva arviointi vaatii sitä, että kurssilla järjestetään harjoituksia, joissa opiskelijat ohjelmoivat itsenäisesti tai pareittain ja opettaja ohjaa heitä tarpeen mukaan. Tällaisessa tilanteessa opettajalla on hyvä mahdollisuus seurata opiskelijoiden ohjelmointia myös ratkaisuperiaatteen muodostumisen osalta. Paitsi ohjaustilanteessa opettajan täytyy luotettavan kuvan muodostaakseen ehtiä keskustella opiskelijoiden kanssa heidän ratkaisuistaan muutenkin. Näin opettaja saa kerättyä runsaasti tietoa siitä, kuinka opiskelija on ongelmaa alun perin lähestynyt ja minkälaisia ajatuksia ja ongelmia ohjelman tuotantoprosessissa on muodostunut.

Havainnointi on teoriassa helppo toteuttaa, mutta käytännössä se sisältää monia vaikeuksia. Ensiksikin on tärkeää, että kurssilla on riittävä määrä sellaisia opetustilaisuuksia, joissa opiskelijat suorittavat itsenäisesti ohjelmointia ja opettajalla on mahdollisuus seurata heidän työskentelyään. Koska tarkoitus on nimenomaan seurata opiskelijoita ja heidän toimimistaan sekä sen kehitystä, täytyy näissä tilanteissa olla tarjolla myös ohjausta. Tilaisuuksien määrä täytyy ehdottomasti määrittää siten, että arvioijalla on riittävä mahdollisuus havainnoida kutakin opiskelijaa. Tämä on ehdoton edellytys havainnoinnin onnistumiselle. Tämä edellyttää toisaalta riittävää määrää tällaisia opetustilaisuuksia ja toisaalta riittävän pieniä ryhmäkokoja.

Havainnointitilanne voidaan toteuttaa joko niin, että opettaja samalla ohjaa opiskelijoita ja tekee havaintoja. Toisaalta tilaisuudessa voisi olla myös ohjaaja tai ohjaajat erikseen, jolloin arvioivalle opettajalle jäisi vain havainnointi. Kummassakin tavassa on hyvät ja huonot puolensa. Jos arvioiva opettaja voi keskittyä vain havainnointiin, pystyy hän käyttämään siihen suuremman osan huomiokyvystään ja siten tekemään tarkempia havaintoja. Toisaalta tällä tavoin arvioiva opettaja ei luontevasti ajaudu keskustelemaan opiskelijoiden kanssa. Kuitenkin opiskelijan ja opettajan välinen keskustelu opiskelijan tuottamasta ratkaisusta ja sen yksityiskohdista on tärkeä osa selvitettäessä opiskelijan ymmärrystä omasta ohjelmastaan ja ohjelmoinnista yleensä. Sopivan informaation kerääminen toisen opettajan ja opiskelijan käymästä keskustelusta voi olla hankalaa ja arvioivan opettajan erilliset haastattelut voivat puolestaan ylimääräisinä häiritä opiskelijan keskittymistä, varsinkin jos häneltä ei samalla tule neuvoja.

Jos opettaja samalla ohjaa opiskelijoita ja tekee havaintoja, jää havaintojen tekemiseen ja niiden analysointiin luonnollisesti vähemmän aikaa ja energiaa kuin puhtaasti havainnointiin paneutuvalla opettajalla. Opettajan on kuitenkin ohjauksen lomassa luontevaa keskustella opiskelijoiden kanssa heidän ohjelmistaan myös varsinaista ohjattavaa kohtaa laajemmaltikin. Hän voi esittää juuri ne kysymykset, joilla kokee saavansa hyödyllisintä informaatiota kyseisen opiskelijan osaamisesta.

Käytännössä isommilla kursseilla ohjaavia opettajia on yleensä useampia. Vaikka arvioija ei toimisi samalla ohjaajana, jouduttaisiin isoilla kursseilla käyttämään joka tapauksessa useampia arvioijia. Tämä voi helposti aiheuttaa epätasa-arvoa, sillä opet-

tajilla voi helposti olla erilaisia näkemyksiä ja ajatuksia oppimistavoitteiden täyttymisestä. Jos arvioijia on useampia, täytyy arviointikriteerit ja oppimistavoitteet käydä todella huolellisesti läpi yhteisesti kaikkien arvioijien kesken. Opettajien henkilökohtaisten näkemysten ja erojen minimointiin tulee kiinnittää erityistä huomiota. Arviointikriteerit ja perusteet täytyy käsitellä erikseen jokaisen arvioitavan kohdan yhteydessä jo etukäteen, jotta kaikki arvioijat tekisivät havaintoja samojen perusteiden pohjalta.

Koska ohjelmoinnin opiskelun havainnoinnissa opettajan ja opiskelijan välisellä keskustelulla on tärkeä rooli, nousee erilaisten inhimillisten tuntemusten merkitys keskeiseen asemaan. Varsinkin jos opettaja sekä ohjaa että arvioi, saavat ryhmän äänekkäämmät jäsenet helposti hiljaisia opiskelijatovereitaan enemmän huomiota. Hyvin syrjäänvetäytyvään opiskelijaan voi olla vaikeaa saada kontaktia, vaikka konkreettista ohjausaikaa olisi paljonkin. Hyvin aktiivinen ja ulospäin suuntautunut opiskelija puolestaan helposti antaa todellista tasoaan osaavamman kuvan itsestään. Opiskelijan henkilökohtaiset tuntemukset opettajaa kohtaan vaikuttavat herkästi heidän välisensä keskustelun sävyyn ja sisältöön. Myös opettajan henkilökohtaisilla tuntemuksilla ja mielipiteillä opiskelijoista on suuri mahdollisuus päästä vaikuttamaan arviointiin.

Näitä kaikkia seikkoja yhdistää yhteinen nimittäjä, henkilön temperamentti. Seuraavissa luvuissa tutustumme tarkemmin tähän kunkin henkilön yksilölliseen ominaisuuksien joukkoon ja sen vaikutuksiin arvioinnissa.

4.3 Arvioinnin muodostaminen

Havainnointiin perustuva arviointi voidaan antaa opiskelijalle yhtenä kokonaisuutena kurssin lopuksi. Tällöin arviointia ei kuitenkaan voida hyödyntää oppimisprosessissa kovinkaan tehokkaasti. Havainnointi ei kuitenkaan millään tavalla poissulje mahdollisuutta erilaisten väliarvioiden antamiseen. Itse asiassa se jopa tarjoaa siihen varsin hyvät mahdollisuudet.

Opettajalla on moninaiset mahdollisuudet antaa opiskelijoille arviointeja myös kurssin suorituksen aikana. Varsinainen kurssiarviointi voidaan jakaa pienempiin osiin, jolloin opiskelijalle annetaan väliarviointeja aina tietyltä periodilta, joka voi olla esimerkiksi jonkin tietyn asiakokonaisuuden käsittelyaika tai tietty ajanjakso. Väliarviot voivat myös olla hyvinkin epämuodollisia, esimerkiksi muutaman sanan lausuntoja opiskelijan suoriutumisesta jossain yksittäisessä harjoitustilaisuudessa. Tällä tavoin opettaja voi helposti myös käyttää arviointia motivoivana ja ohjaavana tekijänä opetuksessa.

Käytännössä havainnointiin perustuva arviointi tuskin voi yksin muodostaa opiskelijan kurssiarvosanan. Kuten luvussa 4.1 on eri kohdissa todettu, on kyseessä hyvin vaikea ja moninaisia heikkoja kohtia sisältävä sekä vaikeasti jälkikäteen todennettava arviointimenetelmä. Nämä seikat sotivat erittäin herkästi arvioinnin peruspilareita, tasa-arvoisuutta ja läpinäkyvyyttä vastaan. Tämä ei kuitenkaan tarkoita, etteikö havainnointi voisi olla hyvä ja käytettävissä oleva arviointimenetelmä ohjelmoinnin alkuopetuksessa.

Vähimpänä vaatimuksena arvioinnin tasa-arvoisuuden ja läpinäkyvyyden toteutumiselle voidaan pitää sitä, että opettaja ja opiskelija käyvät kurssin lopuksi jonkinlaisen arviointikeskustelun. Näin opettaja saa ja hänen myös täytyy perustella opiskelijalle havainnoinnin perusteella antamaansa arvosanaa ja myös opiskelijalla on mahdollisuus kertoa oma näkemyksensä osaamisestaan ja sen oikeuttamasta arvosanasta. Tässä suhteessa ratkaistavaksi ongelmaksi jää se, kuinka toimitaan, jos opiskelijan ja opettajan näkemykset oikeasta arvosanasta eroavat toisistaan, eikä yhteysymmärrystä saada keskustelemalla aikaan. Opettajien käytettävissä olevat aikaresurssit ovat kuitenkin aina rajalliset, eikä keskustelua näin voida venyttää loputtomiin.

Havainnointi voitaisiin myös ottaa osaksi laajempaa jatkuvaa arviointia, joka pitäisi sisällään myös muita tekijöitä. Tällaisia erillisiä arviointikokonaisuuksia voisivat olla esimerkiksi opiskelijoiden kotona tekemät harjoitustehtävät sekä erilaiset itse- ja vertaisarvioinnit.

Havainnointi voidaan joko yksinään tai yhdessä muiden jatkuvan arvioinnin menetelmien kanssa yhdistää tenttien pohjalta tapahtuvaan arviointiin. Tällaisessa kombinaatiossa voidaan käyttää joko peruskoulumaista arviointia, jossa tenttiarvosana muodostaa kokonaisarvosanan perustan, mutta havainnointi tai laajempi jatkuva arviointi voi vaikuttaa tähän joko nostavasti tai laskevasti. Painoarvo eri arviointimenetelmien välillä voisi olla myös erilainen.

Ohjelmoinnin opetuksessa osaamista voidaan testata myös harjoitustöillä, jotka ovat käytännössä yleensä kotitehtäviä laajempien ohjelmakokonaisuuksin laatimista. Näissä perinteisesti suuren eriarvoistavan seikan on aiheuttanut opiskelijoiden valitettava epärehellisyys. Sen, onko opiskelija itse todella tehnyt työn, todentaminen on vaikeaa ja siksi opiskelija voi suhteellisen helposti saada omaa osaamistasoaan huomattavasti paremman arvosanan. Tämä on sinällään valitettavaa, sillä tällainen suurempi ohjelmointityö voi kuitenkin osoittaa opiskelijan todellista ohjelmointiosaamista paremmin kuin parhaalla mahdollisella tavallakaan suunniteltu tentti. Otettaessa havainnointi (tai laajemmin jatkuva arviointi) osaksi arviointia harjoitustyön lisäksi, voidaan tätä epärehellistä arviointia vinouttavaa seikkaa huomattavasti loiventaa ja samalla nostaa kynnystä huijaamiselle aikaisempaa korkeammaksi.

5 HAVAINNOINTIKOKEILU

Kokeilun tarkoituksena oli tutkia, kuinka hyvin opettajat pystyvät arvioimaan opiskelijoiden osaamista pelkän havainnoinnin perusteella ja sitä kautta pohtia, olisiko tällainen arviointitapa mahdollinen ainakin osana kurssin kokonaisarvosanan muodostamista. Tarkoituksena oli nimen omaan tarkastella sitä, millaisen kuvan opettajat muodostavat opiskelijoiden osaamisesta heidän harjoitustilaisuuksissa tapahtuvan suoriutumisensa perusteella. Opettajalle muodostuvaan kuvaan vaikuttivat ainakin epäsuorasti myös opiskelijoiden palauttamat tehtävät ja itsearvioinnit, mutta näitä ei kuitenkaan erityisesti pyritty käyttämään hyväksi arvosanaa arvioitaessa.

Opettajat keskustelivat sekä ennen kurssin alkua että sen aikana kurssin ja erillisten opetustilaisuuksien ja tehtävien oppimistavoitteista, mutta tarkempi arviointikriteerejä tai –taulukoita ei kuitenkaan tehty. Tarkoituksena oli siis arvioida mahdollisimman puhtaasti opettajan inhimillistä kykyä muodostaa arvio opiskelijan osaamisesta ilman tarkkoja ja toteen näytettäviä kriteerejä. Mukana on siis tarkoituksellakin mahdollisuus siihen, että opettaja tietämättään käyttää arviointiperusteena opiskelijan luonnetta, ulospäinsuuntautuneisuutta, sulkeutuneisuutta, epäsosiaalisuutta tms. seikkaa.

5.1 Opintojakso

Kokeilu suoritettiin Ohjelmointi II -opintojaksolla (OHII), jonka laajuus oli 5 opintopistettä. Kyseistä kurssia edeltää 9 opintopisteen laajuinen Ohjelmointi I -opintojakso (OHI). Molempien opintojaksojen opetuskielenä toimii Java. OHI:ssa opiskelijat olivat oppineet ohjelmoinnin perusrakenteet sekä olio-ohjelmoinnin pohjimmaisen idean. OHII keskittyy puhtaasti olio-ohjelmointiin huomattavasti OHI:n olio-osaa syvällisemmin. Lisäksi tutustutaan Javalla kirjoitettuihin valmiisiin luokkiin ja niiden käyttöön mm. laatimalla graafisia käyttöliittymiä.

Kurssi sisälsi 16 luentokertaa (2 * 45 min) sekä 8 harjoituskertaa (2 * 24 min). Tämän lisäksi opiskelijat tekivät omalla ajallaan pieniä ohjelmointitehtäviä sekä vapaaehtoisia itsearviointitehtäviä. Opintojakson lopussa järjestettiin tentti, joka tehtiin

kokonaisuudessaan tietokoneella. Luennoilla käytiin läpi kurssin asiasisältöön liittyviä asioita ja tarkasteltiin näihin liittyviä ilmiöitä esimerkkien kautta. Luennoitsijan omat esimerkit jaettin myös opiskelijoille.

Noin viikkoa ennen harjoituksia opiskelijoille jaettiin harjoitustehtävät, joita tuli tehdä ennen harjoituksia ja palauttaa Moodle-oppimisympäristössä olevaan palautuslaatikkoon. Tehtävistä 4-6 oli ohjelmointitehtäviä ja lisäksi joka harjoituskerralla oli yksi itsearviointitehtävä. Opettajat tarkistivat tehtävät ja merkitsivät ne joko hyväksytyiksi tai hylätyiksi. Hyväksyttyjen tehtävien ei tarvinnut välttämättä olla täydellisiä ratkaisuja, myös puutteelliset ratkaisut hyväksyttiin, mikäli niistä kuitenkin selvästi näki opiskelijan yrittäneen ja pystyneen ratkaisemaan tehtävän riittäviltä osin. Palauttamistaan hyväksytyistä tehtävistä opiskelijat saivat tehtäväpisteitä siten, että kustakin tehtävästä sai kaksi pistettä jos osallistui tehtäviä seuraaviin harjoituksiin ja yhden pisteen jos opiskelija ei osallistunut harjoituksiin. Lisäksi harjoituksiin osallistumisesta sai kaksi lisäpistettä. Läsnäolomerkintä edellytti paitsi läsnäoloa harjoituksissa myös harjoituksiin liittyvän itsearvioinnin tekemistä. Tehtäviä oli tehtävä siten, että opiskelija oli kerännyt vähintään 40% kaikista jaossa olleista tehtäväpisteistä (maksimimäärä on 2 * tehtävien yhteismäärä). Kerätyt harjoituspisteet muutettiin kurssin lopussa tenttipisteiksi siten, että niiden ansiosta opiskelija saattoi saada maksimissaan yhden arvosanan korotuksen saavuttamaansa tenttiarvosanaan.

Tehdyistä harjoitustehtävistä palkitseminen on yleinen käytäntö yliopistotasoisessa tietojenkäsittelytieteen opetuksessa. Hyvityspisteiden tarkoituksena on motivoida opiskelijoita tekemään tehtäviä, mikä selkeästi edesauttaa oppimista ohjelmoinnin tyyppisissä oppiaineissa. Koska kurssi ei ole varsinainen verkkokurssi, opiskelijoille ei anneta hyväksymismerkintää isompaa palautetta Moodlessa. Tämän vuoksi opiskelijoita houkutellaan lähiopetustilaisuuteen isommalla pistemäärällä, jotta he saisivat palautetta tekemistään tehtävistä ja opettaja pääsisi konkreettisesti ohjaamaan heitä. Myös niille opiskelijoille, jotka eivät harjoituksiin jostain syystä pääse, tarjotaan kuitenkin tehtävistä pisteitä, etteivät tehtävät jäisi tekemättä pelkästään siksi, ettei pisteitä ole tarjolla. Tämä selkeästi kvantitatiiviiseen arviointiin perustuva pistesysteemi kuitenkin heijastuu myös kvalitatiivisesti. Korrelaatio tehtyjen tehtävien lukumäärien ja tenttiarvosanojen välillä on yleensä hyvin selvästi nähtävissä.

Harjoituksissa opiskelijoilla oli käytettävissään kurssin johtajan palautettujen tehtävien valitsemia ratkaisuja kotitehtäviin. Opiskelijat kävivät pareittain läpi ennakkotehtävät vertaillen omia ratkaisujaan sekä tutustuen annettuihin ratkaisuihin. Lisäksi harjoitusten pitäjä tarvittaessa auttoi ymmärtämään epäselviksi jääneitä asioita. Tämän jälkeen opiskelijat siirtyivät edelleen pareittain ohjelmoimaan harjoitustilaisuuteen annettuja tehtäviä. Harjoituksissa tehtiin koko kurssin ajan yhtä ohjelmaa, johon lisättiin ominaisuuksia aina kullakin viikolla luennoilla käsiteltyjen asioiden pohjalta.

Opiskelijat oli ohjeistettu kotitehtävien itsearvioinnissa miettimään kunkin palauttamansa tehtävän kohdalla seuraavia asioita:

- Tuntuiko tehtävä helpolta, vaikealta vai jotain siltä väliltä?
- Onko ratkaisu mielestäsi hyvin tehtävänannon täyttävä?
- Mitä opin tai minkä asian ymmärsin tätä tehtävää tehdessäni?
- Minkä avulla sain tehtävän ratkaistua (luennot, aikaisemmat harjoitukset, oppikirja, Internet, tms.)?
- Jäikö ratkaisuusi jotain sellaista, jota et täysin ymmärrä?

Lisäksi opiskelijoiden tuli kommentoida jollain tavalla myös mahdollisesti palauttamatta jääneitä tehtäviä seuraavien kysymysten kautta:

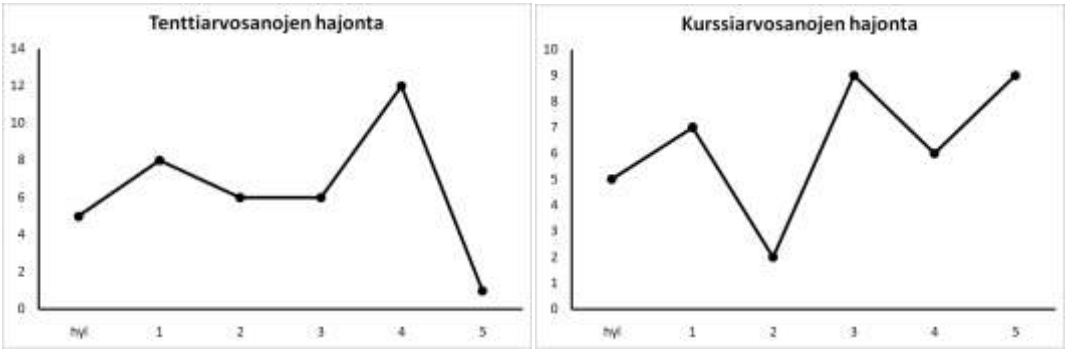
- Mikset palauttanut ko. tehtävää?
- Etkö edes yrittänyt, miksi?
- Jäikö tehtävän tekeminen kesken, miksi?
- Mitä opit yrittäessäsi, mitä jäi ymmärtämättä?

5.2 Osallistujat

Kurssin aloitti 52 opiskelijaa, joista 36 suoritti laskuharjoitukset hyväksytysti ja saavutti siten oikeuden suorittaa kurssin loppu- tai uusintatentillä. Tentteihin osallistui 38 opiskelijaa, joista 33 suoritti tentin hyväksytysti. Tässä tarkasteluun on otettu mukaan kaikki ne opiskelijat, jotka osallistuivat tenttiin riippumatta siitä, olivatko he suorittaneet laskuharjoitukset hyväksytysti. Tentteihin osallistuneista kolme oli sellaisia, jotka eivät olleet suorittaneet laskuharjoituksia hyväksytysti. Mikäli opiskelija osallistui molempiin tentteihin, huomioidaan tässä parempi arvosana.

Kurssiarvosana muodostui tenttipisteiden ja laskuharjoituksista saavutettujen pisteiden yhteismäärästä. Näin ollen pelkällä tentillä saavutetulla arvosanalla ei ole opiskelijoiden kannalta merkitystä, eikä sitä heille erikseen edes ilmoitettu. Koska tentistä saavutetut pisteet ovat kuitenkin täysin laadullisia toisin kuin osittain määrällisesti muodostuneet harjoituspisteet, tarkastellaan tässä myös niitä arvosanoja, jotka opiskelijat olisivat saavuttaneet pelkän tentin perusteella. Tenttiarvosanalla tarkoitetaan siis puhtaasti tenttipisteistä muodostettua arvosanaa, kun taas kurssiarvosana on opiskelijoille kirjattu arvosana, jossa mukana on myös harjoitushyvityspisteet.

Tenttiarvosanojen keskiarvo oli 2,4 ja hyväksyttyjen arvosanojen keskiarvo oli 2,8. Kurssiarvosanojen keskiarvo puolestaan oli 2,8 ja arvosanojen keskiarvo 3,2. Arvosanojen jakautuminen on esitettynä kuvassa 1.



Kuva 1: Opiskelijoiden arvosanojen hajonta

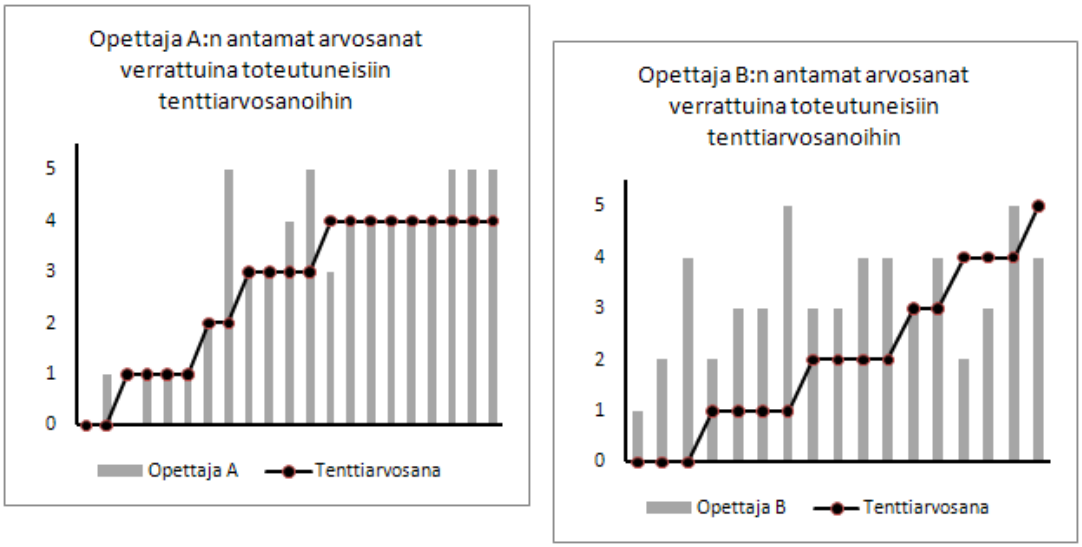
5.3 Opettajien antamat arvosanat

Kurssilla oli kaksi opettajaa. Opettaja A toimi kurssin johtajana ja vastasi näin luennoista sekä harjoitustehtävien laadinnasta. Kurssilla oli neljä laskuharjoitusryhmää joista opettaja A ohjasi kahta ja opettaja B kahta. Opettaja A on toiminut kurssin johtajana vuodesta 2005 alkaen ja tämä oli kyseisellä ajanjaksolla kuudes kurssin järjestämiskerta. Opettaja B on toiminut enimmäkseen verkko-opetustyössä. Hän on sitä kautta hoitanut mm. Ohjelmointi I -kurssia, joka on nyt tarkasteltavan Ohjelmointi II -kurssia edeltävä kurssi. Opettajat esittivät arvosana-arvionsa omien harjoitusryhmiensä opiskelijoista ennen lopputenttiä. Opettaja A tarkasti ja arvioi kaikki tentit.

Tentin suorittaneita opiskelijoita oli opettaja A:n ryhmässä 21 ja opettaja B:n ryhmässä 17. Opettaja A:n ryhmäläiset menestyivät tentissä keskimäärin hieman opettaja B:n ryhmäläisiä paremmin. Opettaja A:n ryhmäläisten tenttiarvosanojen keskiarvo oli 2,7 ja opettaja B:n ryhmäläisten 2,1. Vastaavat keskiarvot kurssiarvosanoille olivat 3,1 ja 2,4.

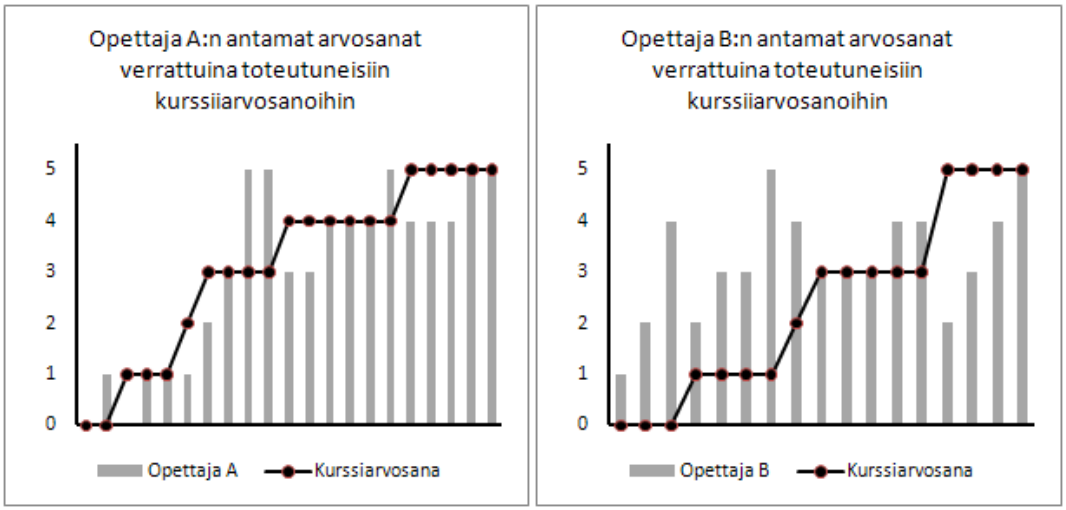
Opettajien antamien arvosanojen korreloinnissa toteutuneisiin arvosanoihin oli melko merkittävä ero. Tämän vuoksi ei ole perusteltua tarkastella opettajien arvioita ja niiden toteutumista yhtenä kokonaisuutena. Tarkastelemme siis kummankin opettajan arvioita erikseen sekä suhteessa pelkän tentin perusteella muodostuviin arvosanoihin että lopullisiin kurssiarvosanoihin.

Verrattaessa opettajan antamaa arvosana-arviota pelkkien tenttipisteiden perusteella muodostettuun arvosanaan, nähdään opettajien välillä huomattava ero. Opettaja A oli arvioinut 12 opiskelijan (57 %) arvosanan samaksi kuin mihin opiskelijan pisteet käytännössä oikeuttivat. Keskimäärin opettaja A:n virhe arvioinnissa oli 0,57 arvosanaa. Opettaja B puolestaan oli arvioinut vain yhden opiskelijan (6 %) arvosanan samaksi kuin hänen tenttiarvosanansa. Opettaja B:n keskimääräinen arviointivirhe oli 1,65 arvosanaa. Molemman opettajat olivat arvioineet useammin opiskelijan osaamista yläkanttiin kuin aliarvioiden. Opettajalla A suhde oli kuitenkin melko tasainen (2/3) kun taas opettaja B oli arvioinut opiskelijoiden osaamisen lähes säännönmukaisesti tentin muodostamaa arvosanaa paremmaksi (13/3). Opettajien arviot suhteessa toteutuneisiin tenttiarvosanoihin on nähtävissä kuvasta 2.



Kuva 2: Opettajien antamat arvosanat verrattuina tenttiarvosanoihin

Verrattaessa opettajan antamaa arvosanaa lopulliseen toteutuneeseen kurssiarvosanaan ero opettajien välillä tasoittuu hieman, mutta pysyi kuitenkin suuntaviivoiltaan samana. Nyt opettaja A:n arvioista 9 (43 %) osui samaksi opiskelijan saavuttaman kurssiarvosanan kanssa keskimääräisen arviointivirheen ollessa 0,67 arvosanaa. Opettaja B:n arvioista 4 (24 %) oli täsmälleen samoja kuin saavutetut arvosanat. Hänen keskimääräinen arviointivirheensä oli kuitenkin edelleen 1,53 arvosanaa. Opettajien arvioiden ja kurssiarvosanojen suhteet löytyvät kuvasta 3.

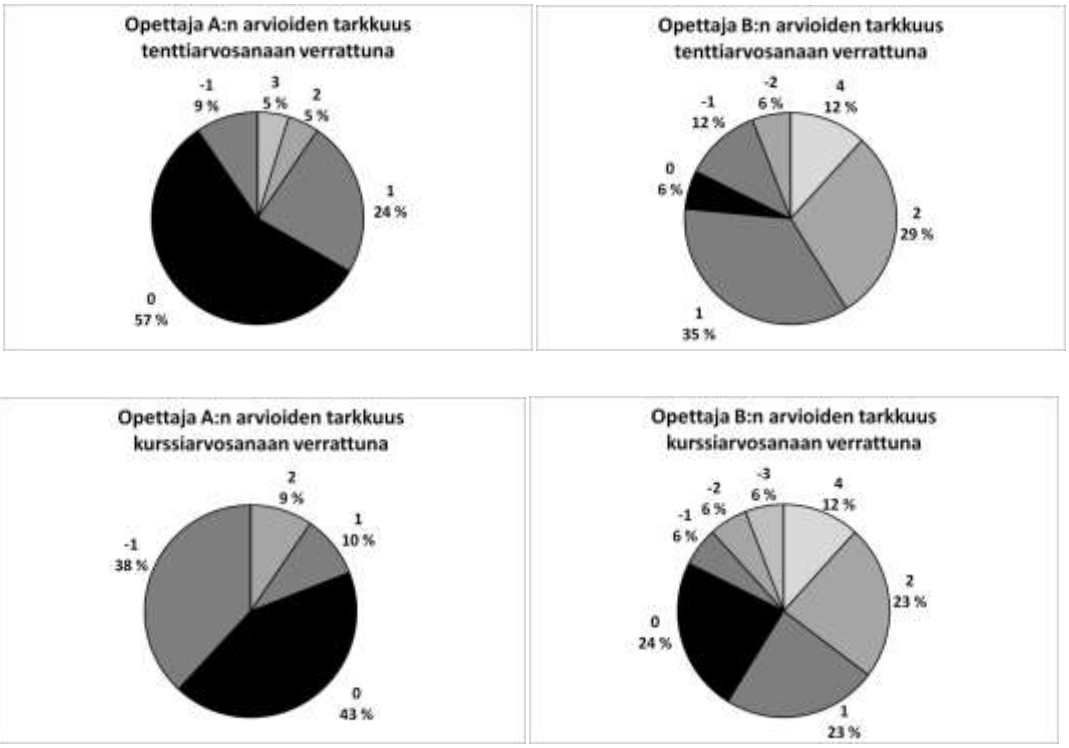


Kuva 3: Opettajien antamat arvosanat verrattuina kurssiarvosanoihin

Tarkastellaan vielä erikseen opettajien havainnointiin perustuvan arvosanan osumatarkkuutta toteutuneisiin arvosanoihin nähden. Kuvassa 4 on esitettynä molempien opettajien arvioiden ja toteutuneiden arvosanojen yhtenevyydet prosenttiosuuksina

kaikista arvosanoista. Myös nämä kaaviot osoittavat hyvin selvästi eron opettajien välillä.

Verrattaessa opettajan arviota toteutuneeseen tenttiarvosanaan opettaja A arvioi osaamisen täsmälleen samaksi yli puolella opiskelijoista ja 90%:lla opiskelijoista hän arvioi arvosanan tenttiarvosanasta maksimissaan yhdellä arvosanalla eroavaksi. Vain kymmenellä prosentilla opiskelijoista opettajan arvion ja tenttiarvosana ero oli kaksi arvosanaa tai enemmän. Opettajalla B vähintään kahdella poikkeavat arvosanat muodostivat 47% kaikista arvosanoista.



Kuva 4: Opettajien antamien arvosanojen osumatarkkuudet tentti- ja kurssiarvosanoihin verrattuna

Opettajan arvion toteutumisessa kurssiarvosanaan verrattuna Opettaja A:n täsmälleen samoiksi muodostuneiden arvioiden osuus pieneni hieman, mutta maksimissaan yhdellä poikkeavien arvosanojen määrä pysyi sunnilleen samana (91 %). Opettaja B:llä täsmälleen samojen arvosanojen määrä kasvoi merkittävästi, mutta kuitenkin kahdella tai useammalla arvosanalla eroavien arvostelujen prosenttiosuus pysyi ennallaan.

5.4 Johtopäätökset

Opettajien antamien arvosanojen hajonnasta voi heti päätellä, että opettajien näkemys opiskelijoiden osaamisesta oli hyvin erilainen, vaikka heidän näkemyksensä kurssin oppimistavoitteista oli ainakin keskustelujen perusteella yhtenevä. Syitä opettajien arviointien eroihin voi olla monia.

Merkittävän tekijän arvioinnissa muodostaa hyvin todennäköisesti se, että opettaja A oli kurssin johtaja ja tentaattori. Vaikka kurssin tavoitteet olikin yhdessä opettaja B:n kanssa käyty läpi, oli opettaja A:lla luonnollisesti vielä tarkempi käsitys siitä, mitä kurssin tentissä lopulta tullaan vaatimaan ja opiskelijan osaamisen vertaaminen tähän näin ollen helpompaa.

Toinen opettajien arviointia eriarvoistava seikka on selkeästi heidän opetuskokemuksensa erilaisuus. Opettaja A on tehnyt lähiopetusta jatkuvasti useiden vuosien ajan, kun taas opettaja B oli työskennellyt pitkään pelkän verkko-opetuksen parissa. Näin ollen hänen rutiininsa tämäntyyppisiin opetus- ja arviointitehtäviin oli huomattavasti vähäisempää.

Edellä mainituista seikoista kumpikin on sellaisia, joita ei voida useinkaan välttää. Kurssilla on aina yksi johtaja ja kurssin sisältö ja oppimistavoitteet ovat viime kädessä hänen määräämiään. Myös opettajien opetuskokemus ja erityisesti kokemus arvioijana voivat vaihdella hyvinkin paljon. Nämä ovat siis seikkoja, jotka on aina huomioitava. Jo tämän kokeilun perusteella on hyvin perusteltua sanoa, että myös käytettäessä havainnointia arviointiperusteena, täytyy arvosteluperusteita tarkentaa huomattavasti, jotta opettajilla olisi mahdollisimman samanlaiset kriteerit arvosanojen muodostamiseen.

Vaikka opettaja A:n arvioinnit näyttävätkin osuneen oikeaan huomattavasti opettaja B:tä paremmin, täytyy muistaa hänen vain onnistuneen arvioimaan paremmin opiskelijoiden toteutuneet arvosanat. Se ei kuitenkaan välttämättä kerro sitä, että hän olisi arvioinut opiskelijoiden varsinaista osaamista paremmin. Tämä kokeilu siis ainoastaan osoitti opettajan kykyä arvioida kuinka hyvin opiskelija selviäisi kurssin lopputentistä. Sitä, kuinka hyvin ko. tentti testasi opiskelijan osaamista, ei todenneta millään tavalla.

Yhteenvetona voidaan siis todeta, että yhtenä erittäin heikkona lenkkinä havainnointiin perustuvassa arvioinnissa voidaan pitää opettajien erilaisia näkemyksiä opiskelijoiden osaamisesta. Tämä seikka täytyy siis huomioida ja pyrkiä eliminoimaan mahdollisimman tehokkaasti, jos havainnointi todella on jossain roolissa opiskelijan arvosanaa muodostettaessa.

6 POHDINTA

Kuten luvussa 4 osoitetut seikat sekä luvussa 5 esitelty kokeilu osoittavat, havainnointia ei ole välttämättä kovinkaan helppoa toteuttaa niin, että se todella palvelee tarkoitustaan, eli antaa totuudenmukaisen kuvan opiskelijan osaamisesta. Erityisesti tasapuolisuuden ja opettajan omien subjektiivisten ajatusten poissulkeminen on haastavaa, muttei mahdotonta. Kurssin opettajien huolellinen oppimistavoitteiden ja arviointikriteerien kartoittaminen sekä jatkuva kommunikointi, sekä huolellinen riskien kartoittaminen ja tiedostaminen vähentävät varmasti virheiden ja epäoikeudenmukaisuuksien mahdollisuutta. Virhearviointien riskiä tuskin voidaan kokonaan eliminoida, mutta tämä pitänee paikkansa kaikissa arviointimuodoissa jossain määrin. Tenttiarvosanakaan ei välttämättä kuvaa läheskään totuudenmukaisesti opiskelijan todellista osaamista kyseisen kurssin aihealueelta, se kertoo ainoastaan hänen kyvystään vastata juuri kysyttyihin kysymyksiin.

Kuten luvussa 2 todetaan, ohjelmoinnin oppiminen on luonnostaan konstruktiivista. Perinteinen tenttiminen ei ehkä parhaalla mahdollisella tavalla sovellu sen arviointiin. Olisi luontevaa, että myös ohjelmoinnin opetus nojaisi konstruktiiviseen oppimiskäsitykseen arviointia myöten. Sinällään havainnointi olisi arviointimenetelmänä tämän suuntauksen mukainen, joustava ja mahdollisuuksien mukaan hyvinkin monipuolinen. Toisaalta ohjelmoinnin syvin olemus, ongelmanratkaisu, ja sen oppiminen hyötyisivät suuresti kognitiivisen oppimiskäsityksen mukaisesta opetuksesta. Tämän mukaiselle oppimisprosessin eri vaiheissa ilmenevälle ja opiskelijan kanssa yhteistyössä tehdylle arvioinnille havainnointi antaisi myös hyviä mahdollisuuksia.

Myöskään perinteisesti ohjelmoinnin oppimisen arvioinnissa vallalla olevat toteava ja kontrolloiva arviointi eivät ehkä ole tarkoitukseen parhaiten sopivat arviointimuodot juuri opittavan asian luonteen vuoksi. Havainnointiin perustuva arviointi antaisi mahdollisuuden ottaa mukaan aineksia motivoivasta, ohjaavasta sekä kehittävästä arvioinnista. Arviointia voitaisiin helposti toteuttaa yhden lopullisen kurssiarvosanan antamisen lisäksi kurssin aikana eri laajuuksissa ja näin valjastaa se myös laajemmin oppimista tukevaan käyttöön.

Kokonaisuutena havainnointi vaikuttaa sopivalta ja mahdolliselta menetelmältä ohjelmoinnin oppimisen arviointiin. Siihen kuitenkin liittyy monia riskejä ja avoimia kysymyksiä, jotka täytyy ratkaista ennen sen käyttöön ottoa.

LÄHTEET

Atjonen, P. 2007. Hyvä, paha arviointi. Helsinki: Tammi.

Finnlex - Valtion säädöstietopankki. Viitattu 8.1.2010. http://www.finlex.fi/

Fuller, U., Johnson C. G., Ahoniemi, T., Cukierman, D., Hernán-Losada, I., Jackova, J., Lahtinen, E., Lewis, T. L., Thompson, D., Riedesel, C., Thompson, E. 2007. ACM SIGCSE Bulletin, 39, 4, 152-170.

Haapasalo, L. 2001. Oppiminen, tieto & ongelmanratkaisu. Joensuu: Medusa-Software.

Itä-Suomen virtuaaliyliopisto 2001 – 2004. Oppimisen arviointi. Viitattu 23.1.2010. http://www.joensuu.fi/isvy/arviointimateriaali/.

JAMK, Ammatillinen opettajakorkeakoulu. Avoin oppimateriaali. Viitattu 23.1.2010. http://aokk.jamk.fi/avoin_oppimateriaali.html

Kari, J. (toim.), 1994. Didaktiikka ja opetussuunnittelu. Juva: WSOY.

Keltinkangas-Järvinen, L. 2006. Temperamentti ja koulumenestys. Juva: WS Bokewell Oy.

Koli, H., Silander, P. 2002. Oppimisprosessin suunnittelu ja ohjaus. Saarijärvi: Hämeen ammattikorkeakoulu.

Opetusalan Ammattijärjestö OAJ. Opettajan ammattietiikka ja eettiset periaatteet. Viitattu 9.1.2010. http://extra.oaj.fi/portal/page?_pageid=515,447767&_dad=portal&_schema=PORTAL

Starr, C. W., Manaris, B., Stalvey R. H. 2008. Bloom's Taxonomy Revisited: Specifying Assessable Learning Objectives in Computer Science. Proceedings of the 39th SIGCSE technical symposium on Computer science education, March 12-15, 2008, Portland, OR, USA.

Thompson, E., Luxton-Reilly, A., Whalley, J. L., Hu, M., Robbins, P. 2008. Bloom's taxonomy for CS assessment. Proceedings of the tenth conference on Australasian computing education, p.155-161, January 01-01, 2008, Wollongong, NSW, Australia.

University of Victoria, Counselling Services. Blooms's taxonomy. Viitattu 23.1.2010. http://www.coun.uvic.ca/learning/exams/blooms-taxonomy.html

Verkkoluotsi. Viitattu 23.1.2010. http://verkkoluotsi.chydenius.fi/.

Wikla, A. 1998. Ohjelmoinnin perusteet Java-kielellä. Hämeenlinna: OtaDATA ry.