Tallinna Pedagoogikaülikool

Informaatika osakond

Jaagup Kippar

Veebiprogrammeerimise terviklik käsitlus

Magistritöö

Juhendaja: Peeter Normak

Tallinn 2002

Sisukord

Sisukord....2
1 Sissejuhatus....5
Aktuaalsus....5
Eesmärgid....5
Lühiülevaade töö sisust....5
2 Veebiprogrammeerimise varasemad õppematerjalid....7
2.1 Ülevaade....7
2.1.1 HTML....7
2.1.2 Javaskript....8
2.1.3 PHP....8
2.1.4 ASP....8
2.1.5 Aruvutivõrk....9
2.1.6 XML....9
2.1.7 VRML....9
2.1.8 PERL....9
2.1.9 Java....9
2.2 Uudisgrupid....12
2.3 Jüri Kiho. Java programmeerimise aabits....13
2.3.1 Algus....13
2.3.2 Kasutajasõbralik ülesehitus....14
2.3.3 Objektid....14
2.3.4 Näited....15
2.3.5 Ülesanded 15
2.3.6 Sõned ja failid....15
2.3.7 Kogumid....15
2.3.8 Mullivann....16
2.3.9 Tagasivaatav rekursioon....16
2.4 Ivor Horton. Beginning Java 2....17
2.4.1 Avastuslugu....17
2.4.2 Algus....17
2.4.3 Tekstipõhine sisu....17
2.4.4 Graafiline kasutajaliides....18
2.4.5 Indeks....19
2.5 Ildar Habibullin. Samoutsitel Java....20
3 TPÜ veebikursus: materjalid ja analüüs....22
3.1 Java põhikursuse konspekt....23
3.1.1 Tutvus programmeerimiskeelega....24
3.1.2 Objektid programmeerimisel....26
3.1.3 Erindid....28
3.1.4 Graafika....28
3.1.5 Vood, failid....29
3.1.6 Internet....30
3.1.7 HTTP server....31
3.1.8 Lõimed....32

3.1.9 Trükkimine....32
3.1.10 Muusika....32
3.1.11 Graphics2D....33
3.1.12 Swing....33
3.1.13 Andmed Javas....34
3.1.14 Dokumenteerimine....34
3.1.15 Installeerimine....35
3.1.16 Tutvustusnäited....35
3.2 Java jätkukursuse konspekt....38
3.2.1 Java3D....39
3.2.2 Rekursiivne joonistamine....39
3.2.3 Maatriksarvutused....39
3.2.4 Joonistamine....39
3.2.5 Pilditöötlus....40
3.2.6 Turvalisus....40
3.2.7 Andmebaas....40
3.2.8 Lühiteemad....40
3.2.9 Muusika....41
3.3 Javaskripti konspekt....41
3.3.1 Pildid....41
3.3.2 Objektid....41
3.3.3 Regulaaravaldised....42
3.3.4 Kihid....42
3.3.5 Andmed....42
3.3.6 Edasiarendusvõimalused....43
3.4 Veebiprogrammeerimise ülesannete kogu....43
4 Näidete koostamise ja esitamise põhimõtted. 45
4.1 Lühikesed näited....46
4.2 Näidete seeriad....47
4.3 Kommentaarid....47
4.4 Näidete valimine....47
4.5 Näidete kättesaadavus....48
4.6 Näiteks lõik töötavat koodi....49
4.7 Näidete paigutamine veebis....50
4.7.1 Dubleerivad viidete lehed....51
4.7.2 Märksõnad....52
4.7.3 Täistekstiotsing....52
4.8 Kokkuvõte....52
5 Õpimudelid ning -teed....52
5.1 Java programmeerimiskeele algõpetus....53
5.1.1 Konkreetsed operatsioonid....53
5.1.2 Käsureamatemaatika....55
5.1.3 Käsureagraafika....56
5.1.4 Mitmekülgsed rakendid....57
5.1.5 JSP põhine õpe....58
5.1.6 Servlettide põhine õpe....58
5.2 Edasijõudnute koolitus....58
5.2.1 Grupiviisiline probleemilahendus....59
5.2.2 Läbipõimitud materjal....59

5.2.3 Mitmetasemelised ülesanded....60
5.3 Veebiprogrammeerimise kursuste komplekte....60
5.3.1 Kliendirakenduste programmeerija....60
5.3.2 Laia profiiliga veebiprogrammerija....61
5.3.3 Andmebaaside veebiliideste programmeerija....62
5.3.4 Veebilehtede koostaja....62
5.4 Õppematerjalide kasutajad....62
5.4.1 TPÜ informaatikatudengid....62
5.4.2 Informaatika kõrvalainena....63
5.4.3 Eesti üldhariduskoolid....63
5.4.4 Arvutifirmad ja programmeerijad....63
5.4.5 Eesti teised ülikoolid ning rakenduskõrgkoolid....64
5.4.6 Abimaterjalid lühikursuslastele....64
5.4.7 Välismaised kasutajad....65
5.4.8 Iseõppijad....65
6 Kokkuvõte....66
Summary....67
Kasutatud allikad....69

Lisa 1: Java põhikursuse konspekt (127 lk)

Lisa 2: Java jätkukursuse konspekt (108 lk)

Lisa 3: Javaskripti konspekt (47 lk)

Lisa 4: Näidete kogu (1000 tk)

Lisa 5: Ülesannete kogu (68 lk)

Lisad on kättesaadavad veebiaadressil http://minitorn.tpu.ee/~jaagup/kool/java/

1 Sissejuhatus

Aktuaalsus

Alates üheksakümnendatest aastatest on veebi osatähtsus arvutimaailmas ning arvutite osa igapäevaelus hüppeliselt kasvanud ning koos sellega suurenenud ka veebiprogrammeerimise tähtsus nii tervikuna kui kogu koodikirjutamisega võrreldes. [51] Ehkki tehnilised võimsused on pea iga kahe aasta järel kahekordistunud, kulub programmeerimisviljakuse kahekordistumiseks enam kui kümme aastat. Seetõttu kipub rakenduste kavandamine ja kirjutamine üha enam kitsaskohaks jääma [51]. Veebiprogrammeerimist puudutavaid kirjutisi ja kursusi on viimaste aastate jooksul loodud mitmeid, kuid enamik neist on keskendunud kitsale teemale või vahendile ning üksikud osad on jäänud omavahel sidumata. Samuti on õppe maht jäänud väiksemaks kui vajadus vastavate oskustega inimeste järele. Kui Eestimaa inimestes tekitada huvi veebiprogrammeerimise vastu ning pakkuda teadmisi ja oskusi, aitab see parandada infotehnoloogilist suutlikkust ning seega ka Eesti konkurentsivõimet maailmaturul; see omakorda muudab siinse elu paindlikumaks ja hõlpsamaks. Käesolev magistritöö püüab seda saavutada.

Eesmärgid

- Analüüsida eestikeelseid ning Eestis kasutatavaid veebiprogrammeerimise materjale ja metoodikaid.
- Kavandada veebiprogrammeerimise õppimiseks terviklik õppekursus
- Koostada loodud kursuse toetamiseks vajalik õppematerjalide kogu
- Piloteerida loodud kursus ning selle põhjal viimistleda metoodika ja õppematerjalid

Lühiülevaade töö sisust.

Magistritöö koosneb analüütilisest osast ning viiest lisast. Kaks esimest lisa on kumbki Java ühe kursuse terviklik konspekt. Kolmas on arvestatud kasutamiseks veebiprogrammeerimise kursuse Javaskripti-teemalise abimaterjalina. Neljandaks on näited, millest peaks abi ja ideid leidma veebiprogrammeerimisega seotud tehniliste probleemide korral. Viimane osa pakub nii õpetajale kui iseõppijale lahendamist väärivaid mitmetasemelisi ülesandeid.

Analüütilises osas antakse ülevaade olemasolevatest veebiprogrammeerimise õppematerjalidest. Lähemalt käsitletakse ning võrreldakse käesoleva magistritöö käigus valminud materjaliga kolme Java õpikut: eesti, inglise ning venekeelset. käesoleva töö autori arvates selleteemalist parimat raamatut. Järgnevalt tutvustatakse koostatud materjalide loomispõhimõtteid ja ootusi peatükkide kaupa. Edasi analüüsitakse näidete koostamist ja esitamist, põhjendatakse praegust valikut ning pakutakse edasiarendusvõimalusi. Siis vaadeldakse koostatud õppematerjalide eeldatavaid sihtgruppe: kellele on loodud konspektid otseselt mõeldud ning kes neist kaudselt kasu saavad.

Mitmeid kordi sarnaseid teemasid õpetades on tekkinud seosed, millised õppetükkide kombinatsioonid paremini kokku sobivad ning millised mitte. Teemade paigutust algõppes olen teadlikult testinud ja jälginud ning õppijate omadusi arvestavad paremini sobivad tervikud on siin välja toodud. Edasi on kombinatsioonid kokku pandud kõrgemal ehk kursuste tasemel ning välja toodud neli edukalt toimivat veebiprogrammeerimise kursuse komplekti.

2 Veebiprogrammeerimise varasemad õppematerjalid

2.1 Ülevaade

Veebiprogrammeerimise materjali on veebis ja trükitud raamatutena ilmunud nii eesti kui võõrkeeltes. On millele toetuda. Hulk vajalikke teemasid on juba kaetud ning õppija võib mitmed veebiprogrammeerimise oskused sealt leida. Käesolevas töös püütakse eestikeelsetest materjalidest anda täielik ülevaade. Võõrkeelsetest tutvustatakse mõningaid siinseid materjale täiendavaid allikaid.

Õppematerjale on koostatud nii lahendatavate probleemide [7][39] kui ka kasutatavate vahendite [8][61][25] kaupa. Lõpuks peab õppija paratamatult jõudma tasemele, kust lahendatava ülesande jaoks sobivad vahendid/keeled leida. Vahendi terviklikku ülevaadet on enamasti kergem anda kui võimaliku lõpmatu hulga veebiprogrammeerimisega seotud probleemide lahendamiseks kasutatavat ammendavat algoritmi. Suurem osa koostatud materjale on vahendikesksed. Selline jaotus paneb õppetöö läbiviijale kohustuse õpetust tasakaalustada ning hoolitseda, et õpilane suudaks teemat haarata laiemalt kui õpikus käsitletud.

2.1.1 HTML

HTML on üheksakümnendate aastate keskelt alates olnud veebiprogrammeerijale hädavajalik selgeks õppida ning neid oskusi kasutatakse veel vähemalt paar aastat, enne kui XHTML või mõni muu keel selle asemele asub [76]. Aastate jooksul on jõutud HTMLi kohta kirjutatud kümnetest eestikeelsetest tutvustustest märkimisväärne osa on väljas NETI lehel [7]. Tartu Ülikoolis on aastaid traditsioonilise õppematerjalina kasutatud Andres Jaegeri lihtsa ja loogilise ülesehitusega konspekti [25]. Tallinna Pedagoogikaülikoolis on veidi põhjalikuma konspekti koos soovitustega kokku pannud Kaja Trees [73]. HTMLi elementide ammendava eestikeelse kirjelduse võib leida [59] Raamatutest kasutatakse [61], Kaido Kikkase lühijuhendit [34] ning Alo Linntamme [39].

CSS-kujunduse poolt sobib lugeda Hans Põldoja kirjutist [52]. Põhjalikuma veebilehtede koostamise, struktureerimise ja kujundamise ülevaate on sama autor andnud oma proseminaritöös [53].

Võõrkeelsed hõlpsamini kasutatavad õppematerjalid on Netscape HTML-guide [4] ning w3schools'i HTML-i osa [77]. Kui tekib keerulisemaid probleeme või arusaamatusi, siis tasub põhjalikumalt uurida w3 konsortsiumi [76] standardit. Samuti võimaldab sealt

ligipääsetav programm (validator) oma lehti standardile vastavuse suhtes kontrollida. Küllalt korralik kirjalik allikas on Danny Goodmani Dynamic HTML [10], kus on lisaks üldisele asjalikule tutvustavale jutule läbi käidud nii HTML-i kui DOMi konstruktsioonid. Keerulisemad elemendid on varustatud asjakohase seletuse ja näitega ning teatatakse, millised seilurid milliste vahenditega paremini hakkama saavad ja kus võib raskusi tekkida. Veebikujunduse osas seletatakse näidete varal, mida veebisaidi loomisel arvestada ning millest hoiduda [9]. Peale üldisi soovitusi on selles raamatus läbi võetud mitut tüüpi veebisaidid (haridus, kaubandus, andmekogu) ning osutatud, kuidas on võimalik selles vallas edukat veebi koostada. Tehnilised üksikasjad on paigutatud lisasse, kuid on seal tihedalt ja arukalt esitatud.

2.1.2 Javaskript

Javaskripti õpetavaid veebilehti ja raamatuid on arvukalt. Selle keele heaks omaduseks on võimalus vaadata töötavatel lehtedel olevat koodi ning siis omi koodilõike koostada – nii nagu HTMLigi puhul. Korralik küllalt terviklik Javaskripti õpetus on koostatud Netscape poolt [44]. Samas leidub ka süstemaatiline Javaskripti dokumentatsioon. Eestikeelsete Javaskripti õpikute autorid on Ilse Sinivee [61] ja Alo Linntamm [39]. Võõrkeelsetest sobib algõpetuseks lugeda Michael Moncur'i Teach Yourself Javascript in 24 Hours [42]. Dünaamiliste veebilehtede koostamisel on heaks abiliseks Nick Heinle Designing with JavaScript [19], milles lähtutakse enamasti kliendi vajadustest, mitte keele loogikast. Lugemise ja järeleproovimise jooksul saab selgeks hulk põhitõdesid ja nendel tuginevaid vajalikke võtteid.

2.1.3 PHP

PHP on praegu tõenäoliselt levinuim keel Eestis lihtsamate ja kesktaseme veebirakenduste koostamiseks. Selle õppimiseks võib leida Kaja Treesi diplomitööna tehtud kirjutise "Õpetusi algajale veebiprogrammeerijale" [74], samuti www.php.ee alt nii õpetusi, soovitusi kui foorumi, kust enesele vajalikku teavet otsida ja küsida. [49]. PHP ametliku dokumentatsiooni ja muude materjalide peegeldus Eestis [50] sisaldab tervikliku käskude kirjelduse koos lihtsate ja selgelt mõistetavate juhenditega. Samuti leiab sealt tarkvara enesele installeerimiseks.

2.1.4 ASP

ASP ehk Microsofti poolt välja pakutud veebiprogrammeerimise tehnoloogia aluseid käsitleb Aulis Sibola diplomitöö [60], mille põhjal on väikeste lisanduste ja suurte

kärbetega koostatud [30]. Sügavuti minnes tuleb leppida võõrkeelsete materjalidega. Mõningast abi võib leida siinse töö autori koostatud näidete kogust (Lisa 4). Kümnetest ASP-d õpetavatest lehekülgedest tunduvad parimad [3] ning [23]. Süstemaatilise dokumentatsiooni ning õpetuse leiab Microsofti leheküljelt [41].

2.1.5 Aruvutivõrk

Veebiprogrammeerimise juures on kasulik mõista ka arvutivõrgu toimimise tööpõhimõtteid ning võimalikke turvariske. Selleteemalise küllalt põhjaliku seletuse võib leida Tiigrihüppe poolt koostatud CD-de Linuxi-osas [79]. Samuti on loengukonspekti koostanud Jaanus Pöial [55] ning teavet leiab veebisaidilt [83].

2.1.6 XML

XML tõotab lähiaastatel saada levinuimaks veebisiseseks andmevahetuskeeleks. Selle alused leiduvad Liivi Haameri proseminaritöös [13], samuti w3 konsortsiumi õpetuste lehel [77]. Põhjalikumalt käsitletakse XSL-i abil andmete muundamist [12].

2.1.7 VRML

Kolmemõõtmelisi töötavaid mudeleid veebilehele annab küllalt hõlpsasti koostada VRML-i abil. Küllalt põhjaliku ning näidetega varustatud õpetuse leiab [82]. VRML-i algõpetuse ja ülevaate mitmetest veebist kättesaadavatest materjalidest koostas proseminarina TPÜs Dimitri Norden [45].

2.1.8 PERL

PERL on aastaid olnud CGI-programmide juures levinuimaks keeleks. Kasutamiskõlblikud algteadmised võib eesti keeles kätte saada Kaido Halliku [17] või Marko Murulaugu õpetustest [43]. Süstemaatiline dokumentatsioon asub keele kodulehel [47].

2.1.9 Java

Java kohta on alates selle avalikust väljakuulutamisest 1996ndal aastal ilmunud mitmeid materjale nii võõr- kui eesti keeles. Esimeseks tuntud õppematerjaliks said 1997. a. sügisel Toomas Lepikult´i koostatud koodinäited Tartu Ülikoolis keemia eriala üliõpilastele programmeerimiskursuse tarbeks [36]. Need näited olid aasta jooksul tähtsaks eestikeelses keskkonnas kasutatavaks baasiks, enne kui ka mujal hakkas Javateemailisi abimaterjale tekkima. Samuti olid näidete juures mitmekülgsed programmeerimisülesanded, mille abil võisid oma teadmisi arendada nii algajad kui

edasijõudnud. Toomas Lepikult'iga õppisime üheskoos 1997. a. kevadel Tamperes Tempus-projekti raames Javakeelset programmeerimist.

1998. a. sügisel luges Tartu Ülikoolis Javapõhist programmeerimise algkursust Jüri Kiho, mille tarvis valmis tal ka eestikeelset terminoloogiat loov ja korrastav Väike Java Leksikon [33]. Mitmed eestikeelsed veebimaterjalid valmisid ka kursusele eelnenud tarkvaraseminari käigus. Küllalt põhjalikult oli nendes käsitletud graafikateemasid, aga 2002. aastal ei õnnestunud neid tarkvaraseminari materjale kahjuks enam veebist leida.

1999. a kevadel pidas Jaanus Pöial Tartu Ülikoolis objektorienteeritud programmeerimise kursuse esmakordselt Java põhjal. Õppureid oli ligi 60 inimest ning mitmele neist on see jäänud meeldiva rahuliku esituse ning asjaliku sisu tõttu parimaks programmeerimisega seotud kursuseks. Ka käesoleva töö autor jäi kuulduga rahule. Õppejõul valmis meeldiv veebileht nii loengumaterjalide, näidete, kopeeritud materjalide kui viidete koguga [54]. Samuti valmisid kursuse käigus mitmed näitlikud programmid ning kasvas peaaegu et välja Javahuviliste ning -oskajate seltskond, kes on valmis saadud teadmisi rakendama ja edasi andma. Väiksemal määral kasutati Java lõike ka teistes TÜ kursustes, nt arvutivõrgud. Järgmistel aastatel on Java keele õpetus TÜs laienenud ning on juurde tulnud nii gruppe kui õpetajaid ning nende poolt koostatud materjale ja ülesandeid. Märkimisväärse osa valminud lehekülgede viiteid võib leida lehelt [54], vastavateemalist õpetust on koordineerinud Jaanus Pöial. 2001. aastal ilmus põhjalikuma käsitlusena TÜs Jüri Kiho koostatud Java programmeerimise aabits [31].

Java levis Eestimaal üha laiemalt ning selle õpetamist alustati nii TTÜs kui IT Kolledžis. Materjale on loonud nii Kaarel Allik kui Tanel Tammet [71]. Enamasti on piirdutud näidete koostamisega või võõrkeelsete kilede ja konspektide kohandamisega. Java õpetuse avalöögi Tehnikaülikoolis tegi Enn Tõugu koos oma Rootsi ja Eesti kolleegidega, pidades intensiivkursuse "Arvutivõrkude programmeerimine Javas". Sealsed materjalid aga olid enamikus veel inglisekeelsed. Eestikeelsed Javaga kokku puutuvad inimesed on lugenud Enn Tõugu selleteemalist küllalt lühikest, kuid tihedat ja terviklikku artiklit "Java fenomen" [75].

Inglisekeelsetest Java-õpetustest on kesksel kohal olnud Java looja Suni poolt koostatud The Java Tutorial [65], pakkudes õpetusi alates lihtsast käivitamisest kuni kesktaseme programmideni. 2000. aasta paiku kippus Tutoriali uuendamine keele arengust märgatavalt maha jääma, kuid viimasel aastal on õpetustega taas järgi jõutud ning

nimetatud kirjutis on taas jõudnud Java õppematerjalide esiritta. Koos The Java Tutorialiga on Sun-i poolt kasutada ametlik Java dokumentatsioon [68] nii käskude süstemaatilise kirjelduse [67] kui korralike seletuste [68] näol. Selle tasuta veebipõhiselt või allalaetavalt kasutatava komplekti põhjal on võimalik usina õppimise abil paljud teemad enesele selgeks teha. Üksikuid tehnilisi probleeme tutvustab lähemalt JDC Tech Tips [69]. Selles võetakse ükshaaval läbi teemad, mille uurimine ja mõistmine vaid dokumentatsiooni abil võib keeruliseks osutuda. Keskeltläbi korra kuus võetakse ette kaks teemat, andes nende üle küllalt tervikliku mõneteistkümneleheküljelise seletuse. Keerulisemaid materjale ning põhjalikumaid seletusi võib leida Sunipoolsest Java arendussaidist [66], samuti standardsete lisapakettide kodulehtedelt [63][64]. Soomekeelsed põhjalikumad käsitlused on [48] ja [80], venekeelseid abimaterjale leiab [70].

Üks korralikumaid ülikoolide veebis välja pandud Java kursusi on Brooklyni ülikoolis [18]. Materjalid on kättesaadavad alates 1997ndast aastast ning neid on süstemaatiliselt uuendatud. Samuti on head vastukaja leidnud Richard Baldwini selges ja loogilises, küllalt sõnaohtras keeles kirjutatud materjalid nii alg-, kesk- kui ka kõrgtasemele [1]. Meeldiv algtasemele suunatud veebiõpik on David J. Eck'i koostatud Introduction to Programming Using Java [6].

Swingi kohta kirjutatud ilusas paarisajaleheküljelises teoses võtavad Matthew Robinsoni ja Pavel Vorobijev läbi nii Swingi enese põhiosad kui sellega lähemalt ja kaugemalt seotud teemad. [56] Raamatukõrvaseid näiteid on teistelgi autoritel välja pandud. Veebiprogrammeerimise seisukohast tähtsamaks võiks osutuda Marty Hally Core Servlets and JavaServer Pages [15]. Samuti on näited kättesaadavad raamatul Java in Nutshell [46].

Näiteid, seletusi ja nõuandeid on kokku kogutud lehtedel [24][2]. Terviklikku Javaportaali on püütud kokku panna saitidel [16][18].

Raamatutest on algusest peale loetud "piibliks" Java in Nutshelli [46], kus küllalt lühidalt kuid selgelt on suudetud tähtsamad teemad kirja panna. Seda raamatut teadsid 1997. aastal soovitada ka Tamperes Javat õpetanud soomlased.

Levinud objektorienteerituse aluseid tutvustav raamat on olnud David n. Smith'i Concepts of Object-Oriented programming [62], kus märgatav osa raamatu sisust on pühendatud põhimõtetele ning illustreerivad lõigud on juurde kirjutatud Smalltalk'i põhjal.

Graafikateemalisi raamatuid on ilmunud mitmeid, millest on tuge saadud nii siinsetel graafikalõikude kirjutamisel kui üliõpilased keerulisemate programmide koostamisel. TPÜ raamatukogust on olnud kättesaadavad [81] ning [20]. Eriti viimases on põhjalikumalt läbi arutatud kahe- ja kolmemõõtmelise graafika nii loogilised kui matemaatilised alused ning antud hulk soovitusi kvaliteetsete graafiliste programmide koostamisel. Edasijõudnutele sobiks TÜ füüsikahoone raamatukogus asuv [40], kus vaadeldakse ka keerulisemaid algoritme nii piltide analüüsil kui sünteesil.

2.2 Uudisgrupid

Nii algajal kui edasijõudnul on võimalus abi leida nii uudisgruppide arhiividest, jälgida sealset jooksvat vestlust kui selles ise osaleda. Suhteliselt kõige lihtsamaid veebiprogrammeerimisalaseid probleeme käsitletakse uudisgrupis ee.arvutid.www, kus enamik programmeerimisalaseid teateid piirdub HTMLi või Javaskriptiga, kuid kust leiab lahendusi väga mitmesugustele veebiga seotud probleemidele. Enim otseselt veebiprogrammeerimisega seotud teateid tuleb gruppi ee.arvutid.www.webmasters, kus põhiliselt PHP ja MySQLi-ga kuid täiesti märgatavalt ka muude veebitehnoloogiatega seotud probleeme arutatakse. Eraldi eestikeelseks programmeerimisgrupiks on ee.prog, kus programmeerimisteemasid sügavamalt lahatakse. Ehkki märkimisväärne osa arutlusi pole otseselt veebiga seotud, leiab sealt veebiprogrammeerimist õppija küll ja küll kasulikku, mida tal ka veebiprogrammide koostamisel vaja läheb. Võrreldes välismaiste listidega on siinsed grupid küllalt rahulikud - keskmiselt neli-viis kirja päevas, kuid samas on ettevõetavad teemad just meie kultuurikeskkonnas tähtsaimad ning vaidluste korral võib kirjade hulk kergesti mitmekordistuda. Küllalt tihedaks ja asjalikuks uudisgrupiks on comp.lang.java.programmer, kust võib leida kirjutisi nii otse Java keele kohta kui ka muudel veebiprogrammeerimise teemadel. Ülejäänud lähedaste teemade uudisgrupid on märgatavalt lahjema sisuga, kuid spetsiifilisi lõike võib neistki leida. Andmebaasidega seotud probleemidele võib lahendusi otsida grupist comp.lang.java.databases, kasutajaliidese koostamise üle arutatakse grupis comp.lang.java.gui. Nii konkreetse probleemi kui ka üldisemate põhimõtete puhul on sageli mõistlik enne keerulisemate raamatute otsimist uudisgrupid üle vaadata, kus võib küllalt kerge vaevaga lahenduse leida.

2.3 Jüri Kiho. Java programmeerimise aabits

Tegemist on kahtlemata parima ja põhjalikema eestikeelse paberil ilmunud Java õppematerjaliga. "Käesolev programmeerimise aabits on mõeldud eeskätt abimaterjaliks õppeaine Programmeerimine (MTAT.03.100) omandamisel", väidab autor. Teos on mõeldud kasutamiseks koos õppejõu seletuste ja nõuannetega ja nii täidab see oma ülesande. Nüüd on tudengitel võimalik järele vaadata, kui tunnis midagi segaseks jäi. Varem samalt autorilt ilmunud Väike Java leksikon aitas küll korrastada keelelisi mõisteid, kuid sobis pigem hilisemaks kordamiseks ja teemade meeldetuletamiseks kui uue õppimiseks.

Et tegemist on esimese eestikeelse sellealase õpikuga, siis on tema sihtgrupp palju suurem kui pelgalt Tartu Ülikooli esimesel kursusel programmeerimist õppivad tudengid. Minu magistritöö raames koostatud õppematerjal ja Jüri Kiho õpik taotlevad lähedast eesmärki. Kirjutised täiendavad teineteist, kattuvaid lõike peaaegu pole. Sama näiteks ei saa öelda mitmete ingliskeelsete raamatute kohta. Sunderlandi Ülikooli raamatukoguriiulis läbi vaadatud paarikümne raamatu hulgas leidus palju kattuvaid või äravahetamiseni sarnaseid näiteid ja seletusi. Kogu suure hulga seast võis ehk leida kolme raamatu jagu erinevat materjali.

Siinse õppematerjaliga võrreldes on Jüri Kihol vähem teemasid: keskendutakse ühele kursusele ning ülejäänud ainest käsitlevad põhjalikumalt juba teised õppejõud oma kursustes. Julgesti on kasutatud matemaatilisi mõisteid ja teemasid, mis siinses õppematerjalis on suurelt osalt jäetud vaid jätkukursuse sisse. TPÜs õpetamisel saadud kogemused on näidanud isegi matemaatika eriala üliõpilaste (rakendus)matemaatika-pelgust ning seetõttu on võimaluse korral püütud näiteid mitte liialt matemaatilisteks ajada, vaid on püütud hakkama saada õppurite arvates vähem arusaamatusi tekkivate teemadega, et informaatikaalased põhiteadmised ei jääks selle tõttu omandamata. Eelkirjutatu ei tähenda, nagu kursuses eneses arvutusoskust ja põhi- ning keskkooli matemaatikat vaja ei läheks. Mõnikord keskmistele ning sageli nutikamatele üliõpilastele antud ülesannete puhul on paratamatu kokku otsida aastate jooksul kogunenud teadmised nii aritmeetikast kui geomeetriast ning nende abil loovaid lahendusi leida ja tööle panna.

2.3.1 Algus

Kui tahta arvestada ja aidata raamatuga ka iseõppijat või esimesel tunnil puudunud üliõpilast, siis võib Jüri Kiho aabitsa esimeses vihikus ettevõetud näide liialt keeruline

tunduda. Iseenesest pole seal midagi keerulist, kuid ehk oleks lihtsa tervituse stiilis abinäide veidi pehmemaks vahelüliks uustulnukale programmeerimise alustamiseks. Ka seletused on loogilised ja asjalikud, kuid tundmatute mõistete tulv võib muuta teksti kergesti arusaamatuks ning kinnistada rahva seas levinud arvamust, et igasugune programmeerimine on vaid piiratud kildkonnale arusaadav toiming, mille juurde tavainimesel asja pole. Algajale on esimese peatüki üle kahekümne uue mõiste liig mis liig. Edasistes peatükkides on neid alla kümne, isegi mahukas ekraanigraafika vihikus vaid seitse eraldi lõpus välja toodud mõistet (tegelikkuses on seal siiski veidi rohkem meeldejätmist ja arusaamist vajavaid kujundeid ning vahepeal märgitud mõisteid). Kui üldhariduskoolis (näiteks füüsikas) omandatakse keskmiselt kuni kolm uut üksust või seost tunnis, siis ei tohiks ka ülikoolis eeldada, et üle kümne sõna suudetakse mõtestada ühe korraga. Nagu aga enne mainitud, raamatu statuudiga siin vastuolu pole, sest koolis õpikuna kasutades on õppejõul õigus, võimalus ja kohustus probleemsed kohad pikemalt lahti seletada ning näitlikustada. Alustamine on probleemiks pea iga õpiku ja kursuse juures, et kirju taustaga õppurkond suudaks väikese tasanduse järel üheskoos ning ühes suunas liikuma hakata. Ehk ainukesena on suudetud programmeerimise mõte ja skeem algajale selgeks teha sama autori raamatus "Kuidas programmeerida" [29], kus on aga selle teema peale paarsada lehekülge kulutatud.

2.3.2 Kasutajasõbralik ülesehitus

Raamatus on leitud tähelepanu ja ruumi väikeste, kuid algajale koodikirjutajatele arusaamatute lõikude seletamiseks nagu näiteks reataanded (vihik 2, lk. 9). Nii saab tähelepanelik lugeja vajalikud teadmised muuseas kätte, ilma, et ta seda eraldi õppima peaks. Samuti teeb lugemise meeldivamaks autori julgus kasutada meetodite ja muutujate nimedes täpitähti. See võib küll mõne konfiguratsiooni juures tekitada probleeme kompileerimisel või käivitamisel, samuti võivad pildi kirjuks ajada Windowsis tavaredaktorite ning DOS-i käsurea erinevad kooditabelid, kuid Tartus enamlevinud Unix-masinate puhul ei tohiks täpitähed raskusi valmistada ning koodi keeleline pilt paraneb tunduvalt.

2.3.3 Objektid

Aabitsa algupoolel on sisse toodud objektid ja pärimine. Mõne autori arvates võib tegemist olla liigse kiirustamisega. Näiteks Tampere Tehnikaülikooli Pori osakonnas hoiti objektidest eemale terve esimese semestri jooksul. Minu kogemused näitavad, et alustades

varakult objektidega, tunduvad need õppijatele loomulikud ning hiljem ei teki salapärast hirmu tundmatu valdkonna ees.

2.3.4 Näited

Jüri Kiho raamatu suureks plussiks on terviklikud käivitamiskõlbulikud näited. Õppimisele aitab kaasa võimalus näited Internetist alla laadida, et veenduda programmi töökõlbulikkuses. Nii saadakse üle mõnikord ületamatuna tunduvast raskusest, kus üksiku viltuläinud märgi peale annab kompilaator arusaamatu veateate ning võõra teksti korrektuur ei taha kuidagi töövõimelist tulemust anda.

2.3.5 Ülesanded

Vihikute lõpus olevad ülesanded pakuvad õppijale võimaluse oma oskusi proovile panna. Uue teemaga tutvumisel võib enesele jõukohase ülesande välja mõtlemine küllalt raskeks osutuda, sest esiteks pole ideid, mida kasulikku õpitud vahenditega peale hakata ning teiseks ei suudeta hinnata toimingute keerukust ning vastavate programmide loomiseks vajatavaid lisateadmisi.

2.3.6 Sõned ja failid

Võib tunduda üllatav, et õppematerjalide hulgas on tutvutud tekstipõhiste jadafailidega enne kui sõnetöötluskäskudega, kuid koostatud näidete taustal paistab selline järjekord täiesti loogilisena. Ehkki sõnetöötlus võib algajale tunduda arusaadavam kui failidega majandamine, annab see tagapool võimaluse käsitleda ka veidi keerulisemaid sõnedega seotud käsklusi. Samuti teeb varasem lihtne tutvus failidega nendega läbikäimise loomulikumaks ning ei teki vahel ette tulevat hirmu failide kui seletamatu ja ohtliku nähtuse ees.

2.3.7 Kogumid

Kogumite juures on teema rahulikult seletatud juba Java algsest versioonist pärit klassi Vector abil, mille eeliseks on, et ei pea muretsema oma kasutatava keskkonna versiooninumbri pärast. Ehkki Java arendajad on soovitanud tarvitada samaks otstarbeks klassi ArrayList ning vahelepaigutavate andmete korral LinkedListi, siis Vectori põhjal on võimalik kõik toimingud selgeks õppida ning hilisem nimevahetus kuigivõrd õppimist ei nõua. Kui olemasolevad vahendid rahuldavad, siis pole vaja iga hinna eest uute juurde tormata, kui need edasiantavale sõnumile suurt juurde ei anna. Pea samamoodi oli ühes

finantsprobleeme käsitlevas programmeerimisõpikus põhjendatud Java versiooni 1.0.2-ga piirdumist, "because it works".

Mõnevõrra harjumatu on klaviatuurisisendi ja -väljundi juures kasutada loodud abiklassi SV. Iseenesest on tegemist sama lähenemisega kui Rein Jürgensoni Programmitehnika [28] graafikaosas, kus vajati pea alati lisaks abifail kaivita.pas. Ka enese tarbeks programme koostades olen selliseid abivahendeid loonud ning näiteks Microsoft on oma J++'i luues rõhutanud konsoolilt lihtsustatud sisendi-väljundi keele märgatavat eelist. Samas olen - eriti algajate õpetamisel - avastanud, et iga juurdetoodud tükk võib kellelegi komistuskiviks saada.

Kui pole just taotluslik näidata klassi Integer isendi loomist ja sealt int-väärtuse pärimist, siis kannataks ehk klassi Kogum real 27 kirjutada (new Integer(r)).intValue() asemele lihtsalt Integer.parseInt(r). Mõni selline vanem kirjaviis kipub ka omal sisse ununema. Meeldivaks vaadata ning kasulikuks läbi mõelda on meetodite skeem, millisest mida välja kutsutakse. Selline joonis aitab programmi struktuuri selgemaks saada ning samas pakub lugejale võimaluse ja harjumuse oma programmi kirjeldamiseks.

2.3.8 Mullivann

Koos rakenditega on siin vihikus tegemist ainukeste graafikat käsitlevate osadega kogu aabitsas ning nende jooksul on läbi võetud pea kõik tähtsamad graafilise kasutajaliidese koostamisega seotud ideoloogiad ja käsklused. Kes on end koostatud näidetest läbi töötanud ning veidi muutusi läbi katsetanud, sellel ei tohiks tulevikus enam lihtsamate graafiliste programmide koostamisega probleeme tekkida. Julgesti on kasutatud veidi uuemaid ning juba küllalt hästi omaks võetud Swingi graafikavahendeid. Kuularitega tutvumist on alustatud akna sulgemisest, mis sobinuks ka minu töösse. Kui eelmises vihikus oli meetodite omavahelise suhtlemise skeem, siis siin on klasside, isendite ja teadete saatmise skeem, mis samuti aitab teadmisi korrastada ning süvendab korraliku kirjeldamise harjumust.

2.3.9 Tagasivaatav rekursioon

Viimaseks vihikuks sattunud rekursiivsed meetodid on tagasivaatav samm graafiliste ilustuste juurest traditsioonilise programmeerimise juurde. Arukalt ja rahulikult on defineeritud nii ülesanne, algoritm ja meetod ning sealtkaudu jõutud rekursiivse algoritmi mõisteni. Näidete juures on alustatud kõigile mõistetavatest leivalõikamisest ja puuraiumisest ja jõutakse ka nupukamatele vähemasti esimesel korral peamurdmist

nõudvate Hanoi tornideni. Nii nagu mina jõuan oma materjalide puhul failisüsteemi või teineteisesse sisenevaid pilte analüüsides rekursioonini, on ka siin tegemist täiesti loogilise teega.

2.4 Ivor Horton. Beginning Java 2

2.4.1 Avastuslugu

Aukartustäratav teos - seda nii sisu kui mõõtmete poolest! Edinburghi raamatipoes oli 2000. aasta kevadel rida Javale pühendatud teoseid nii algajatele, edasijõudnutele kui ka üksikute teemade süvendatud käsitlusi. Algõpetust sisaldavatest raamatutest jäi Beginning Java 2 huvitava ning samas sõbralikuna silma. Suur osa raamatu sisust oli tuttav. Uurisin sealsamas läbi enesele uudse muusikaosa ja tutvusin raamatu ülesehitusega. Nädala pärast Londonis oli sama raamat hea ja verivärskena särava iseloomustuse abil esile toodud. Hiljem sain Tallinnas IT-koolituse inimestelt raamatule kiitva hinnangu. Otsustasin nimetatud raamatu omakoostatud materjalile võrdluseks valida.

2.4.2 Algus

Ehkki on tegemist ka algajatele mõeldud teosega, kipuvad esimesed leheküljed selles suhtes veidi konarlikud olema. Ilusti seletatakse eesmärke ja põhimõtteid, samuti raamatu ning näidete kasutamist, kuid esimene kasutajapoolne programmi käivitamine tuleb justkui muuseas teiste toimingute vahel alles raamatu kuueteistkümnendal leheküljel. Nii nagu Java 2 Complete [5] raamatus, nii ka siin on esimeseks tööle lükatavaks programmiks rakend. Rakendi puhul on rakendikäituri töö eesmärk õppijale kergemini mõistetav. Valmiskompileeritud programmide käivitamiseks vajaliku interpretaatori ülesanne kipub Pascali või C abil .exe faile loonud inimesele mõttetu raiskamisena ning samas häirib seekaudu ka õppimist. Lihtsa programmi käivitamine esimese peatüki lõpus võetakse veel korra natuke teisel kujul läbi. Nii on loota, et ka algajad iseõppijad võiksid sellega omal käel hakkama saada.

2.4.3 Tekstipõhine sisu

Järgmises peatükis on küllalt julgesti ette võetud mitmesugused andmetüübid ning nendega sooritatavad tehted. Varasema programmeerimiskogemusega inimestele ei tohiks need kuigi palju raskusi valmistada, kuid ka algajatel peaks heade seletuste abil olema võimalik pakutud teadmisest aru saada ning seda soovitud kohtades rakendada. Iga

peatüki lõpus on ülevaatlik kokkuvõte ning parajalt jõukohased harjutused, samuti vajadusel viited andmete juurde hankimiseks. Kuna on tegemist mitte liialt mahuka materjaliga, siis on üles kirjutatud pea kõik ohud ja erijuhud, mis õppejõud muul juhul loengus juurde räägib. Kirjutamise stiil on lahe, sugugi mitte robustne. Hea kujutlusvõime korral võib ette kujutada, et ollakse meeldivas loengus, kus on võimalik lehe keeramise kiiruse abil tempot määrata.

Ainsa Java-raamatuna minu nähtutest on siin massiivid ja sõned seletusele võetud samas peatükis. See on küll tavaks olnud C õpikute puhul, kus eraldi sõnetüüpi polnudki, kuid paistab ka siin edukalt õnnestuma. Sellise ühendamise puhul annab edukalt seletada ühisosi ning samas välja tuua erinevused. Nii selles peatükis kui mujalgi on küllalt palju meetodite kirjeldusi, mida põhimõtteliselt on igaühel võimalik dokumentatsioonist vaadata. Kui arvutit ei juhtu käepärast olema, siis on selline lähenemine abiks. Samuti pole häbenetud sama lõiku või joonist korrata, kui selleks peaks vajadus tekkima. Ise olen püüdnud lehekülgede arvu kokkuhoiu mõttes kordumisi siiski võimalikult vältida, ehkki mõnikord lihtsustab õigesse kohta uuesti kopeeritud lõik mõistmist, kuna võimaldab sujuvalt lugeda ning ei sunni otsitud koha vaatamiseks lehekülgi lappama.

Objektidega seonduv on kahe peatüki jooksul süstemaatiliselt läbi võetud. Isegi paketid on nime poolest algajate raamatusse paigutatud, kuid rahulikud ja loogilised seletused peaksid nende vajaduse ja kasutamise lugejale arusaadavaks tegema.

Failid ja vood on ühte peatükki kokku pandud nagu minulgi, ehkki voo abil kirjutamine või failinimede vaatamine on põhimõtte poolest küllaltki erinevad tegevused. Mitmesugused tarvikud on kokku pandud ühte pikka peatükki, ilma et neil omavahel kuigi suurt seost oleks. Kuna igaüht on eraldi küllalt terviklikult kirjeldatud, siis selline ühendamine ei häiri. Võib juhtuda, et edaspidistes raamatutes võivad ka failide ja kataloogidega manipuleerivad käsud hakata kuuluma eraldiseisvate tarvikute alla, sest võrgunduse arenedes tunduvad üha loomulikumatena ka failidega sidumata vood.

2.4.4 Graafiline kasutajaliides

Raamatu esimene pool on piirdunud kasutajaga suhtlemisel vaid tekstiekraaniga nagu traditsioonilistele programmeerimisõpikutele kohane. Järgneva viiesaja lehekülje jooksul aga võetakse põhjalikult läbi nii graafilise kasutajaliidese koostamine kui piltide loomine ja töötlemine. Teemad on loogiliselt järjestatud ning nende lugemine on nauditav. Arvutil järgi proovides tekib usk, et kõik ka tegelikult nii käib kui kirjas. Üldjuhul on

osatud hoolitseda nii, et järjest lugedes on kõik tarvilikud mõisted eelpool seletatud. Graafika vahele on pikitud ka sellega seotud teemasid, näiteks ajaarvutused ning meetodite perioodiline käivitamine animatsioonide puhul.

Kirjeldatav raamat on pea esimeseks kohaks, kus koos JDK 1.3-ga valminud Sound API-le on korralik loodud näidetega õpetus. Varasemad teadmised tuli kõik postiloenditest või siis napisõnalisest dokumentatsioonist kokku otsida. Rohkem kui seitsmekümnel leheküljel on läbi võetud nii põhimõtted kui lihtsamad ja veidi keerulisemad heli esitamise, salvestamise ja töötlemise viisid. Kui olin võrgust otsitud seletuste ning enese muusikalise ja füüsikalise hariduse põhjal jätkukursuse muusikalise osa kokku pannud, oli tagantjärele sealt hea vaadata, kuidas teine autor samu teemasid lahendab. Jällegi on temal kirjas muul juhul loengul räägitav jutt ning näidatavad joonised. Samuti on teemad läbi mängitud, nii et lugejal tekib kasutusvaldkondadest ja -võimalustest mitmekülgne ettekujutus. MIDI näited olid peamiselt üles ehitatud klaviatuurile, sügavamatesse muusikalistesse arutlustesse selles raamatus ei laskutud.

Lõpupeatükkides võetakse ette andmebaasid - ikka samad teemad, millest pea üheski põhjalikumas Javat tutvustavas õpikus ei pääse. Ning nagu selles raamatus kombeks, paigutatakse mujal käsitlemata jäänud, kuid sisuliselt tähtsad lõigud põhiteema vahele. Andmebaaside juures on graafilise kasutajaliidese loomise näitel põhjalikumalt seletatud Swingi JTree toimimist. Mitmed eraldiseisvad teemad ja andmed leiduvad lisades: jar-failid, dokumenteerimine, võtmesõnad, ASCII koodid jm.

2.4.5 Indeks

Väga vajalik ning kasutajasõbralik on Beginning Java 2 raamatus olev indeks - peatükk, mis annaks väärtust juurde nii minu koostatud materjalidele kui ka Jüri Kiho raamatule. Isegi raamatu autor ise ei suuda alati kiiresti leida lehekülge, kus ta otsitud teemal on kirjutanud, rääkimata tavakasutajast, kes raamatu ehk vaid korra on kaanest kaaneni läbi lugenud või tahab lihtsalt soovitud teema kohta kiiresti midagi järele vaadata. Loogiliselt paigutatud ja pealkirjastatud peatükkide ning alateemade kaudu on küll võimalik kergendada soovitud kohtade leidmist, kuid paratamatult on teemad vähemalt osalt niivõrd läbi põimunud, et kõiki soovitud kohti pole võimalik pealkirjade järgi üles leida. Ivor Hortoni raamatus on indeks lausa kaheks jagatud: programmikäsud ning teooriateemad eraldi. Piisavalt suure indeksi puhul selline jaotus hõlbustab otsimist ning teeb niigi hea raamatu veelgi nauditavamaks.

2.5 Ildar Habibullin. Samoutsitel Java

Tegemist on kõige konkreetsema ja asjalikuma venekeelse Java õpikuga, mida siinse magistritöö autor kohanud. Raamat pole teiste omasugustega võrreldes paks – alla 500 vähem kui A4 formaadis lehekülge, kuid hulga sisukam kui mitmed suurema mahuga raamatud. Joonistega pole liialdatud, kuid hädavajalikumates kohtades on nad väikeselt ja selgelt olemas. Lihtsa programmi käivitamine on kohe esimese sisulise peatüki alguses. Juurde on lisatud ligi kaks lehekülge kommentaare, mis on muu teksti lühidust arvestades suhteliselt palju. Viimane aga näitab, kui tähtsaks peetakse esmase kokkupuute selgust. Edasi võetakse traditsioonilisel teel läbi nii andmetüübid, keelekonstruktsioonid, objektid kui paketidki. Ei häbeneta tähelepanu pöörata ka keerulisematele objektorienteeritusega seotud valdkondadele (sisemine klass). Samas on seletused nii selged, et ka varem programmeerimisega vähe kokku puutunud nupukam alustaja peaks suutma seletatut mõista.

Edaspidises kirjelduses jätkatakse sama liini. Nii sõnetöötluse, mähisklasside kui ka utiliitide juures tuuakse sisse klasse ja käsklusi, mis muudes raamatutes paigutatakse jätkukursuse alla. Samas on näited suudetud valida nii elulised, et kogenum programmeerija suudab käigu pealt haarata, millega tegu ning algajagi ei tohiks kolmekordse ülelugemise ning järeleproovimise peale hätta jääda.

Graafika juurde asutakse raamatu esimese kolmandiku lõpus ning sealgi minnakse pärast lühitutvust juba Graphics2D vahendite juurde. Graafilised programmid on algselt tehtud Frame alamklassina, alles hiljem rakendite juures tuuakse sisse võimalus, et programmi saab käivitada nii seilurist kui käsurealt.

Keerulisem graafika ning heli koos MIDIga on paigutatud vähem kui kolmekümneleheküljelisse peatükki. Selle läbitöötamine annab oskused, mis võinuks muul juhul sadu lehekülgi kirjeldusi ja kümneid näiteid võtta. See osa eeldab ka kogenenumalt lugejalt juba suuremat pühendumust, kuid tundub, et midagi tähtsat pole ütlemata jäänud, mida õppija ei suudaks ise välja mõelda.

Tagapool on asjatundikult läbi käidud nii lõimed, vood, failid, võrk kui andmebaas. Igast teemast nopitud välja tähtsam ning kergemalt mõistetavam osa, nii et kasutajal jääb üle vaid edaspidi dokumentatsiooni uurida ning ta võib varsti keerukaid rakendusi mõista ja mõningase kogemuse tekkimise järel ka neid kokku panna.

Sellise ülesehitusega raamat kuluks igakeelsele lugejale. Eriti mugav on seda kasutada õppetöö abimaterjalina, kus on võimalik raamatus käsitlemata jäänud segased teemad lõpuni selgeks rääkida ning antud põhja peale omapoolne analüüsiv rakendus ehitada. Kuid nii nagu pealkiri ütleb, eelkõige on "Samoutsitel Java" ikkagi mõeldud süvenemisvõimelisele ja kohusetundlikule iseõppijale.

3 TPÜ veebikursus: materjalid ja analüüs

1999. a. sügisel pidas käesoleva töö autor TPÜs esimest korda veebiprogrammeerimise kursuse. Eesmärgiks võeti koostada siinsete õppurite jaoks vähemalt sama sobiv kursus, kui selle oli TÜ jaoks loonud Jaanus Pöial. Et enamik õppijaist olid tulevased arvatavad pedagoogid, ei võinud silmaringi koomamale tõmmata. Lisaks tuli tervikliku pildi saamiseks sisse võtta teemad (nt. Javaskript), mis TÜs või TTÜs leidsid käsitlemist eraldi ainetena. Eesmärkide saavutamiseks polnud võimalik olemasolevate materjalidega piirduda, vaid tuli koostada uus sobiv õppematerjal/konspekt. Otsustati luua olemasolevatele napisõnalistele märksõnalistele materjalidele kõrvale võimalikult ladusalt jutustav tekst, mida annaks mõista ka ilma õppejõupoolseid seletusi kuulamata ning ilma laialdasi teemakohaseid eelteadmisi omamata. Otseseks tõukeks igaks loenguks konspekti loomiseks sai vastavateemaline lubadus TPÜs õppivale pimedale noormehele, kes suutis arvutis olevat teksti lugeda.

Sellelt kursuselt jäi Javaga edaspidi tihedamalt seotuks Charles Roos, kes järgmisel aastal koostas suvepraktika raames Java algõpetust läbi viiva programmi "Java alused" [57] ning hiljem ise servlettidepõhist Java algkursust pidades koostas materjali ja metoodika taolise õpetuse tarbeks, kaitstes selle ka proseminaritööna [58].

Materjali koostamisel ning kirjutistesse jagamisel kaalus käesoleva magistritöö autor mitmeid võimalusi. Uues pedagoogikas soovitatakse probleemipõhist lähenemist, mõnes raamatus keskendutakse algoritmidele, mõnel pool aga lähtutakse kirjutamisel keele vahenditest. Mitmekuulise võimaluste kaalumise ning võrdlemise tulemusena leidis autor, et õppijale kasulik oleks läbi viia praktikumid probleemipõhiselt, loengud aga keskendada algoritmidele ja konkreetse keele pakutud võimalustele, võimaluse ja vajaduse korral kõrvaljooksvaid seoseid kommenteerides. Sellisel juhul jääb praktikumitundides aega avastamiseks ning isetegevuseks, loengumaterjali poolt aga tuleb kindel skelett, millele on pärast kergem toetuda ning ettetulevale probleemile hõlpsamini lahendus leida või avastada lahenduse puudumine. Selle tagamõttega saidki loodavad konspektid ja näited jagatud keelte ja tehniliste vahendite järgi, ülesannete kogu aga koostatud valdavalt probleemipõhisena.

Kõikide loodud konspektide põhjal on ka tegelikkuses õpetatud. Java põhikursuse konspekti peatükke on kasutanud paarikümne grupi õppurid, Java jätkukursuse konspekti järgi on õppinud kaks gruppi ning Javaskripti materjalid on läbi õpitud seitsmel korral.

Igal puhul on õppejõule jäänud ka tagasiside kas siis kirjalikus, suulises või kognitiivses vormis. Õppijaiks on olnud nii koolilapsed, üliõpilased kui täiskasvanud.

3.1 Java põhikursuse konspekt

Loodud konspekt (Lisa 1) on mõeldud nii iseõppijatele kui ka abimaterjaliks õpilastele ja õpetajatele. Kasulikke soovitusi või ideid peaksid siit leidma nii algajad kui ka kogenumad koodikirjutajad. Iseõppijailt eeldatakse varasemat kasvõi väikest kokkupuudet programmeerimisega, kuid lisamaterjalide või kõrvalise abi toel peaks olema võimalik selle materjali abil ka päris algusest alustada. Teoreetilise suunitlusega inimesed saavad hakata esimesest peatükist alates lugema ning teemad on püütud nii järjestada, et saab liikuda lihtsamalt keerulisema poole.

Kel aga on soovi kohe näpuga järele proovida, kuidas midagi tööle panna ja edaspidi juba töötavat programmi oma soovide ja huvide järgi kohandada ning sealtkaudu õppida, siis võib julgesti alustada raamatu lõpuosast. Installeerimise juures on seletustega piltkirjas ära toodud etapid, mille abil on võimalik programmi koostamiseks vajalikud vahendid Internetist enesele salvestada, oma masinas töökorda sättida ning nende abil lihtne programm koostada ja tööle panna. Järgnevate tutvustusnäidete juures alustatakse sellest samast lühikesest programmist ning lisatakse tasapisi võimalusi, püüdes siiski programm jätta lühikeseks, lihtsaks ja kergesti mõistetavaks. Kes on need näited mõttega läbi lugenud, läbi proovinud, väikseid ja suuremaidki muudatusi sisse viinud ja edukalt eri programmide osi ühendada suutnud, need peaksid hakkama saama mitmete tavaelu lihtsate abiprogrammide koostamisega ning samas on suutnud enesele ehitada aluse, kindlustunde ja usu endasse, millele edasisi keerulisemaid teadmisi kinnitada. Ka koolitunnis või arvutiringides on soovitav alustada lihtsatest töötavatest näidetest, millega mängides ei käi programmeerimine üle jõu. Näited pakuvad õpetajale 1) küllaldaselt võimalusi tagamaade seletamiseks (mida võib kasvõi siitsamast eelnevatest peatükkidest lugeda); 2) lasevad nii nõrgematele kui tugevamatele jõukohaseid ülesandeid jagada; 3) tutvuda teooriateemadega. Mõistmise sügavus lihtsalt oleneb õppimisvõimest ning pühendumusest, kuid lühikese valmisnäite juures suudab igaüks juhendaja juuresolekul midagi muuta või täiendada.

3.1.1 Tutvus programmeerimiskeelega

Keelega tutvumise peatükis on seletatud olulisemad käsud ning toodud mõned näited, mille abil on võimalik edaspidi palju tarvilikke ning tunduvalt asjalikumaid programme kokku panna. Kel varem mõne programmeerimiskeelega kogemusi, see peaks keele peatüki abil saama aimu enam ette tulevatest käskudest ja konstruktsioonidest ning suutma traditsioonilises programmeerimises ette tulevad arvutavad ja kirjutavad programmid kokku panna. Sedasorti programmid võivad küll keerukamaks minna algoritmide poolest, kuid keeleliselt on kõik oluline olemas, mille abil saab ka näiteks olümpiaadidel ette tulevaid ülesandeid lahendada. Sisendist on jäetud vaid käsurea parameetrid, mida on kõige lihtsam kirja panna. Kui piisab, et kasutaja saab programmile algandmed anda, siis neist parameetritest täiesti jagub. Kasutaja poolt jagatavate andmete puhul on enamasti võimalik kogu teave algul ette anda ning siis pärast arvutamist tulemust vaadata. Faili sisend-väljundit ning klaviatuurilt vooga sisendit käsitletakse voogude ja failide peatükis. Need on teemad, mis üldjuhul ka tekstipõhiste programmide juurde käivad.

Sõne juurde on pealtnäha loogikavastaselt sisse toodud StringTokenizeri abil lausete tükeldamine - teema, mida teiste programmeerimiskeelte õpikus üldjuhul nii algusse pole nihutatud. Kuna aga tegemist on tekstipõhiste programmide juures küllalt sageli ette tuleva operatsiooniga, siis on lause tükeldamise näide koos seletustega esitatud siin; osalt ka seetõttu, et sõnede tarvis mujal terviklikke lõike pole koostatud ning nii on kasutajal parem otsida ühest kohast. Kui tulevikus käsitleda sõnesid laiemalt, siis võiks ka lause tükeldamine olla eraldi vastavas peatükis koos StringBufferi, regulaaravaldiste ning muude sõnetöötlusega seotud vahenditega.

Arvutamise juures on tutvustatud klassi Math käsklusi, lisaks ka täisarvulise juhuarvu loomist. Enamasti suudavad tudengid, olles nende vahenditega tutvunud, kõiksugu valemid ilma suuremate raskusteta kirja panna, kui nad on jõudnud need juba paberil oma programmi tarbeks välja mõelda.

Abiinfo tutvustamine aitab loodetavasti otsida soovitud käsklusi ning õpetab kirjapandut mõistma. API dokumentatsiooni lugemine on algul paljudele küllaltki vaevaline toiming, kuid paari klassi käskluste lähema läbiuurimise järel suudetakse paremini mõista, mida ülejäänud klassidegi omadustelt oodata on.

Struktuursed andmetüübid nõuavad algul rohkem läbimõtlemist, kuid nendest ei saa suuremate programmide puhul enam üle ega ümber. Varem programmeerimisega

tutvunutele ei tohiks vähemasti massiivid kuigi suuri raskusi valmistada; esimesi kordi nende poole pöördudes aga tuleb ise hulgaliselt harjutusi läbi teha, et sedapidised lahendused tuttavate ja usutavatena tunduksid ning nende abil oleks tahtmist ja julgust uusi ja keerulisemaid rakendusi kokku panna. Näitena on sisse toodud ka kolmemõõtmeline massiiv, et kahemõõtmeline ei tunduks kasutajale keerukuse tipuna, vaid et ta näeks, et tegemist on ühtlase süsteemiga ning julgeks vajadusel vähemalt kahemõõtmelise omadusi ilma pelguseta tarvitada. Siia peatükki on paigutatud ka klassi mõiste ja lihtne omakoostatud andmetüüp: Pascalit õppinutel tekiks seos kirjega ning selguks, et ise ühe tüübi kokku panemine ei olegi kuigi keerukas. Tegu on objektorienteeritud lähenemisele pinna ette valmistamisega. Samas lõpetatakse peatükk kohas, kus autor arvab, et oleks kasulik neil enesele piir tõmmata, kes tahavad tavaliste programmide koostamisega hakkama saada, kuid ei taha objektidega seotud peensustesse süveneda ning selle peale suuremat aega kulutada. Klassi sees oleva alamprogrammi kasulikkuse näitamiseks on ka konstruktor siia peatükki sisse võetud, kuid samas ei ole tegeldud isendimeetodite välja kutsumisega ehk teadete saatmisega. Julgesti on küsitud ja paigutatud isendite väljadele väärtusi. Ehkki see on objektorienteeritud mõttelaadiga vastuolus, saab siin veel piirduda Pascali/C kirjete mõttemalliga, kus väljade väärtuste pärimiseks ja määramiseks ei kasutata meetodeid. Kuna konstruktor on ühekordseks väärtuste omistamiseks, siis ei hakka ka see tekitama taunitud segatud olukorda, kus osa andmeid väärtustatakse töö käigus meetodite, osa otsese kopeerimise abil.

Lühidalt on läbi mängitud olukord, kus kaks osutit näitavad samadele andmetele ning selle tulemusena avaldub ühe osuti poolt põhjustatud muutus ka teise osuti kaudu küsitud väärtustes. Ehkki siin on näide lühidalt kirja pandud, tasub tunnis teemat läbi võttes või ka ise materjali uurides see lähemalt läbi mõelda ning arutleda, millised järeldused sellel on ning mis erinevused on lihttüüpide ning struktuurtüüpidega programmeerimisel.

Nagu mõne aasta pikkused kogemused näitavad, omandatakse ja mõistetakse selline algne süntaks küllalt kergesti, eriti kui ollakse varem mõne keelega (TPÜ puhul Pascaliga) tuttavad. Pikemat aega küll aetakse segi termineid (kiputakse nii ploki, tsükli kui valiku kohta ütlema tsükkel), kuid sisuliselt saadakse aru millega tegu, ning osatakse vahendeid õigesti kasutada, mis ongi ehk kõige tähtsam. Samuti kipub segamini minema, et Pascalis on sõnede liitmiseks (sidurdamiseks) koma (tegelikult funktsiooni parameetrite eraldaja), Javas aga plussmärk. Kui aga kompilaator järjekindlalt teatab, mis õige mis

vale, siis pärast paari esimest segadusekorda tuleb kirjutajal juba ise meelde milles asi, ning edasisi programmikoostamisi erinevus enam oluliselt ei häiri.

Keelega tutvumise peatükis tundub sobimatu olevat arvusüsteemide lõik, kuna algsel omandamisel kipub ta olevat liialt abstraktne ning mõistetamatu. Vähemasti mitmed kogemused algajatega on näidanud, et kahend-, kaheksand- ja kuueteistkümnendsüsteemi vajaduse mõistmine õnnestub edukamalt alles siis, kui neid ka tegelikult vaja läheb. Et aga vastavad teemad on aastaid kuulunud programmeerimisse [37][36], on nad ka siinses õppematerjalis paigutatud algtaseme muude oskuste juurde. Seoses informaatika nihkumisega rangelt matemaatikaga seotud teadusharust pigem (multi)meediumi poole tuleb tõenäoliselt vastav lõik tulevikus viia andmetega seotud peatükki.

3.1.2 Objektid programmeerimisel

Peatükis on püütud seletada ning näidete põhjal läbi proovida objektorienteeritud programmide koostamisel kasutatavad lõigud, pannes tähtsamad teemad ettepoole ning harvem ette tulevad taha. Objektorienteerituse mõistmisel peaks olema kergem aru saada nii programmide ülesehitusest kui kuvatavatest veateadetest. Samas peaks olema raamatu materjale võimalik mõista ning programme kokku panna ka ilma siinset peatükki läbimata, kuid sel juhul tuleb arvestada, et programmide kirjutamine põhineb mitmelgi juhul analoogial ja intuitsioonil ning koodi kirjutaja pole võimeline alati põhjendama, miks tema loodud rakendus nii käitub, nagu see välja paistab.

Lihtsaima klassi loomine on paigutatud juba eelmise peatüki lõppu struktuursete andmetüüpide alla, siin minnakse teemat pidi põhjalikumaks. Peatüki algul kirjeldatakse objektide kasutamise peamisi vajadusi. Põhjalikumad eestikeelsed seletused selle tarvis on olemas. Siin on ühe objektiga küllalt lihtne näide ning kohe asutakse objektidega seotud järgmiste teemade juurde. Õppides aga tasuks sellelgi teemal pikemalt peatuda ning püüda ise mitmeid klasse kavandada, koostada ja katsetada.

Pärides jätkatakse eelmise näite pealt ning lisatakse sellele vaid üks meetod ja lihtsuse mõttes mitte ühtegi välja. Kui edaspidi on põhjust õpikut edasi arendada, siis oleks tõenäoliselt vajalik ka välja lisamine ette võtta. Eelmine näide oli nii koostatud, et sellele oleks suhteliselt lihtne teha alamklassi: sinna oli paigutatud teise konstruktori kõrvale ka parameetriteta konstruktor, mis alamklassi isendi loomisel vaikimisi välja kutsutakse. Saanuks ka piirduda sootuks ilma konstruktoriteta klassidega, kuid autorile

tundus, et praegune algväärtustamisvõimalusega lahendus on parem. See hoolitseb, et ei tekiks ebasoovitavat tava väljadele ilma meetodita väärtuste omistamisel. Isendimuutujaid pole küll private abil väljapoolt nähtamatuks muudetud, kuid seda pigem koodi lühidust silmas pidades. Lühemat koodi on tunduvalt kergem haarata - nagu olen mitmel juhul tähele pannud - ja iga üleliigne sõna või rida toob juurde inimesi, kellele just vastava rea tõttu mõistmine märgatavalt raskemaks muutub.

Pärimisel tuleb juurde võimalus alamklassi isendit ülemklassi tüüpi osutile omistada ning pärast algse objekti erimeetodite kasutamiseks tarvitada tüübimuundust. Vaid iseõppijaile mõeldud materjali puhul oleks siia tõenäoliselt mõistlik vähemasti üks sellekohane kommenteeritud näide lisada, kus ülemklassi osutist tüübimuunduse teel alamklassi isend saadakse, kuid siin on piirdutud vaid sõnalise kommentaariga ning muundus ja alamklassi meetodi käivitamine sulgude abil lühiduse huvides ühte lausesse paigutatud. Eraldi on koostatud näide samast klassist pärinevate isendite paigutamiseks ühte massiivi; see näitab võimalust muutujate hulka vähendada ning eriomadustest loobuda, kus selleks otsest tarvidust pole.

Ülekatte ja liideste tarvis on koostatud omaette näidete perekond. Eelmises seerias kasutatud mitmesuguste punkti omadustel põhinevate näidete edasiarendusena oleks samuti võimalik olnud nende juurde minna - mõnedes õpikutes on ka sama aluse pealt pea kõik toimingud seletatud - kuid siin otsustati omaette grupp luua, et oleks võimalik näiteid lühikestena ja kergelt haaratavatena hoida. Nagu õpetamisest saadud kogemused näitavad, saavutati nende peatükkide puhul soovitud tulemus.

Piiritlejate ja lisaoskuste sissetoomisel peatutakse põgusalt. Koostatud materjalide põhjal tunde pidades on võimalik sihtrühmaga arvestada ja vajadusel teemad põhjalikumalt läbi arutada ja näidete kaudu proovida, kuid õpikus on - vähemalt praeguses versioonis - teemadest küllalt kergesti üle libisetud, püüdes sealjuures tähtsamad pidepunktid anda, mille külge edaspidisi kogemusi kinnitada.

Sama lugu on ka sisemiste klassidega. Paari näite kaudu peaks nende kasutamine mõistetav olema, kuid peenemaid ette tulevaid olukordi ning kaudsemaid võimalusi õpikus läbi ei mängita. Komponendid ja oad on jäänud sisse lihtsalt tutvustavate lõikudena.

3.1.3 Erindid

Erindid on väike teema, kuid samas vajalik töökindlate programmide loomiseks. Lähem kirjeldus ja soovitused asuvad Java jätkukursuse konspektis. Siinsete tekstide läbitöötamise tulemusena peaks kasutaja paremini mõistma tekkivaid veateateid ning suutma adekvaatselt reageerida vastavalt vea tüübile. Võiks tekkida üldine ettekujutus, mille tarvis on valmistüübid loodud ning kus tuleks enesel hakata uusi erinditüüpe looma.

3.1.4 Graafika

Graafikaosas on toodud välja vaid põhilised vahendid ja seletused. Esmase kokkupuute võiks kasutaja saada tutvustusnäiteid uurides. Siinsed tekstid on rohkem mõeldud võimaluste tutvustamiseks ning põhimõtete seletamiseks. Kuna aga tegemist on küllalt tähtsa valdkonnaga, siis edasiarenduse puhul tuleks mitmed siinsed teemad ning komponendid pikemalt lahti kirjutada, et ka neil õppuritel oleks võimalus näpuga järge ajades mõistetava ja kasuliku tulemuseni jõuda, kel õpetaja jutust või üksikute näidete nägemisest ei piisa.

Paigutamine on Javas näiteks võrreldes Visual Basicuga küllalt keeruline ning siin pole liialt põhjalikuks mindud teadlikult, et koodikirjutajad ei hakkaks kujundust toimimise kõrval liialt tähtsaks pidama.

Kuularite puhul on püütud jätta põhiline toimemehhanism läbipaistvaks ning arusaadavaks. Alustatase ühest lihtsast ning mõningase mõtlemise peale mõistetavast näitest ning siis liigutakse edasi mitmesuguseid kuulareid tutvustades. Nende abil peaks olema võimalik enamik tavarakendustes ettetulevaid ideid realiseerida. Samuti peaks vaadatud kuularite abil olema mõistetav ka teiste dokumentatsioonist loetavate kuularite eesmärk ning toimimispõhimõte. Tulevikus oleks tõenäoliselt vajalik kirjutada näitena isekoostatud teadete saatja (ümmargune nupp), samuti sündmusi laiali jagav isend (paaris- ja paarituarvulised hiirevajutused) koos korraliku skeemiga, et lugejal oleks lähem ettekujutus sündmuste tekkimisest ja liikumisest. Hulknurka näitav graafikakomponent on kokku pandud ning selle abil saab mitmed tähtsamad teemad ja toimimismehhanismid läbi vaadata.

Joonistamise ning liikumisega seotud teemad tuleks tulevikus tihedamalt läbi võtta. Jätkukursuse seas on küll veidi graafikapõhimõtteid kirja pandud ning samuti on koostatud liikumist illustreeriv järkjärgult arendatud sirelasemäng, kuid ammendatuks ei saa teemat veel pidada. Arvestades, et võõrkeeltes on sel teemal kirjutatud mitmeid

vajalikke raamatuid, kuluks meilgi sellele teemale veidi rohkem pühenduda. Asjalikud seletused leiab siiski Jüri Kiho raamatu 13. peatükist (mullivann), samuti on tähtsamad graafikaga seotud lõigud käsitletud TÜ tarkvaraseminaride materjalides (mida küll 2002. kevadel kahjuks ei õnnestunud enam veebist leida).

3.1.5 Vood, failid

Voogude ja failide juures on põhjalikumalt ette võetud Java andmevahetusskeem. Käsitletakse voogude muundamist ning baidi- ja tekstivooge. Alustatakse loodetavasti küllalt lihtsalt mõistetavate käskluste ja põhimõtetega DataInputStream'ist ja FileInputStream'ist. Eeldatakse, et peatükki lugema asudes on pärimisega seotud põhimõtted selged ning need aitavad omistamiste mõistmisel. Kuigi tekstis püütakse rõhutada Java mitmekihilist andmeedastusskeemi, kus üldjuhul alumine kiht tegeleb andmetega vaid üksiku baidi/baidimassiivi tasemel, kuluks õpikusse ometigi ära tähelepanu tõmbav võibolla isegi karikatuurne joonis, mis suudaks selle teadmise kinnitada õppija mällu tähtsa ning säravana.

Toru on toodud sisse vaid tutvustamise mõttes. Et lõimedega tegeldakse lähemalt tagapool, siis edasiarenduses sobiks sinna põhjalikumalt lõimedevahelist torude abil andmevahetust tutvustav traditsioonililne tootja-tarbija mudel. Usinamate otsijate jaoks on küll täiesti korralik selleteemaline näide olemas Jaanus Pöidla loengumaterjalides [54]. Samuti kuluks reaaleluga seostatav näide vooge ühendava SequenceInputStreami juurde.

Juhupöördusfaili on samuti vaid tutvustatud. Kui mõnes (eriti C ja Pascali) kursuses kulutatakse märkimisväärselt vahendeid tutvustamaks fikseeritud suurusega kirjefailidega opereerimist, siis siin on püütud jääda voogudepõhiseks. Samas on tegemist siiski küllalt tähtsa teemaga, mis vajaks lähemalt käsitlemist jätkukursuses.

Failide ja kataloogide andmete pärimine erineb küllaltki palju failide sisu käsitlevast osast ning suurema lahtikirjutusastme korral nõuaks omaette peatükki. Siiski võiksid nad siin veel eelmainitud osaga kokku sobida. Toodud on paar lihtsat kommenteeritud näidet, mille mõistmisel on võimalik märkimisväärne osa failidega ette tulevatest ülesannetest kombineerimise teel lahendada.

Alamkataloogide hierarhiasse tungivad kommenteeritud näited võib lihtsama kursuse puhul vahele jätta. Samas need pakuvad edasijõudnutele küllalt näitlikku ning

arendavat mõtlemisainet ning loodetavasti äratundmist, et samu võtteid võib olla kasulik tarvitada päris paljudes valdkondades. Samuti tundub failisüsteemi abil rekursiooni tutvustamine vähemalt siinsele autorile palju mõttekam ning elulisem kui faktoriaali puhul, kus hariliku iteratsiooni kaudu on ülesannet silmnähtavalt tuttavam lahendada. Ülesande lahendamiseks kõrvutatakse kaks teed, mis kasutavad sama tulemuse saamiseks kaht täiesti erinevat loogikat; mõlema põhimõtte järgi on võimalik enamik selliseid operatsioone nõudvaid ülesandeid lahendada. Peab lihtsalt suutma ülesande taandada vastavale mudelile ning siis kaalutleda, milliseid ressursse millises koguses kasutada on. Suurema ühevärvilise pinna leidmiseks graafikas näiteks rekursioon enamiku keelte ning interpretaatorite puhul ei sobi, sest üksteisesse sisenevaid alamprogramme tuleb nii palju, et arvuti ei suuda neid välja kanda. Andmete salvestamiseks on küll näites senitundmatu vahendina sisse toodud nimistu, mille kasutamine peaks paarikordse läbilugemise järel täiesti mõistetav olema. Samuti peaks kord kohatud nimistut kasutav näide soodustama ka tagapool tuleva Java Collections Framework'i vahendite mõistmist.

3.1.6 Internet

Selles peatükis on tasapisi keerukamate lahenduste juurde minnes püütud lihtsate näidete varal seletada võrguprogrammide töötamispõhimõtteid. Pärast teema läbitöötamist peaks kasutaja mõistma võrguprogrammide ehitust ja ettetulevaid probleeme ning suutma koostada interaktiivseid klient-server rakendusi.

Võrgu tehniliste külgede juurde pole konspektis mindud, seda on siinse taseme programmide koostamise oskamiseks juba eesti keeles mitmel pool tutvustatud (viited). Samuti on esitatud vaid võrguprogrammide koostamist tutvustavates kirjutistes toodud järeldusi ilma sügavamatesse arutlustesse laskumata.

Alustatakse ehk lihtsaima ning kergesti mõistetava veebiprotokolli - kellaaja küsimise - kasutamisega, püüdes kinni ning trükkides ekraanile serverist saabunud rida. Voogude peatükist saadud teadmised peaksid aitama siin seoseid luua ning kirjapandut mõista ning rakendada. Kahepoolse ühenduse näiteks on toodud finger-teenus, samuti lihtne ning kergelt/hõlpsasti/ mõistetav protokoll.

Küllalt kiiresti liigutakse serverprogrammide ning sealtkaudu ka lõimede juurde. Praktilisema suunitlusega kursuse puhul võib nendel teemadel pikemalt peatuda ning mitmeid valmislahendusi kokku panna. Lõimed tunduvad võrguprogrammide juures loomulikuna ning siin on nende seletamine läinud libedamalt ning tulemuslikumalt kui

näiteks liikumise juures lõimedega alustades. Näide, kus tuleks Kadrioru sularahaautomaadi juures pikalt oodata, kuni keegi Raplas oma arveldustega ühele poole saab, paneb igaühe mõistma, et suuremate süsteemide puhul võib paralleelprogrammeerimine hädavajalik olla. Selline äratundmine aga suurendab vähemalt mõneks ajaks õpimotivatsiooni ning pealtnäha küllalt raskest lõimede teemast saadakse mõnekümne minutiga algne arusaamine ja loomisoskus kätte. Edasised nüansid ja eriolukorrad võivad õppimise ja tegutsemise indu kärpida, kuid tekkinud põhja peale on järgmistel tundidel juba kergem uusi teadmisi ning konkreetseid rakendusi ehitada.

Õppijate imestust äratab lihtsalt kokkupandav jututuba ning loodud põhja peale ehitatavad tekstipõhised või graafilised kestad. On juhtunud, et graafilise võrgurakenduse kokkupaneku näiv lihtsus on meelitanud õppureid mitmeid graafikateemadest segaseks jäänud lõike enesele selgeks tegema. Näitena lisatud Trips-Traps-Trulli mängu põhjal on täienduste abil võimalik juba märgatav kogus võrguprogrammide tarbeks tekkivaid ideid realiseerida. Peatüki lõpetab veebipõhine sünkroniseerimist vajav oksjoni näide. Teoreetilisemas kursuses on selle põhjalt võimalik seletada märkimisväärne osa ühiste ressursside kasutamisel tekkinud probleemide lahenduste teid ning nende võrdlusi ja realisatsioone.

3.1.7 HTTP server

Nimetatud peatükk võiks eelmisega peaaegu et kokku kuuluda - tegu on lihtsalt Interneti osa jätkuga - siin aga vaadeldakse vaid üht osa sellest. Kui eespool alustati protokolli loomist nullist, siis siin eeldatakse juba töötavat veebiserverit ning uuritakse, mis sellega seondub. Suur osa piirdub tutvustava jutuga ning on toodud paar illustreerivat servletti. Samal teemaga seotud on ka jätkukursuse konspektis mõned lõigud, kuid teema vajaks põhjalikumat läbikirjutamist. Lähemalt tegeldakse servlettidega Charles Roosi [58] kursuse ning proseminaritöö raames, kuid arvestades mitmetes keeltes selleteemalisi terviklikke raamatuid, oleks vähemasti iseõppija tarbeks ja õpetajate koolitamisel vaja nii mitmekülgsemaid kommenteeritud näited, andmebaasiteooriaga seotud lõike kui turvalisusega seotud kirjutisi. Vähemasti lähiaastatel tundub veebiserverite programmeerimine üha hoogustuvat ning paljud suuremad süsteemid on valinud aluseks Java platvormi. Mitmed loodud kirjutised (Alo Linntam, Kaja Trees, Aulis Sibola) on suutnud algtaseme tarbeks küllalt häid materjale kokku panna. Eesti infotehnoloogiaga

maailma esirinnas püsimiseks kuluksid ka keerulisemad õppematerjalid marjaks ära. Mingist tasemest alates tuleb tõenäoliselt nagunii võõrkeelsete uudismaterjalidega piirduda, sest kõiki lihtsalt pole mõistlik tõlkida. Vähemalt mõned kümned leheküljed selleteemalist tutvustavat ning analüüsivat teksti kuluksid selles valdkonnas maakeelsena ära. Need aitaksid ka eesti keelt korrastada ja säilitada ning oleksid hüppelauaks veebis ja mujalgi leiduvatele võõrkeelsetele materjalidele.

3.1.8 Lõimed

Õppija tutvus teemaga varjatult ning küllalt suurel määral võrguprogrammide juures. Siin aga püütakse anda lõimedekohane teoreetiline taust ning seletada võrguvaldkonnast eemal olevaid lõimi vajavaid olukordi. Liikumisnäitena toodud pallide põrkamine sarnaneb küllaltki Jüri Kiho mullivanniga, kuid on vähemate võimalustega ning loodetavasti selle tõttu ka lihtsam mõista. Ootamise juures on sisse toodud monitori/luku mõiste ning tutvustatud, kuidas selle abil on võimalik tööjärg soovitud sündmuseni (näiteks nupuvajutuseni) peatada.

Võib juhtuda, et edasiarenduse korral on teema mõistlik üle viia jätkukursuse materjalide hulka, võttes põhjalikumalt läbi nii lõimede ja andmete paralleel- ning järjestiktöötlusega seotud probleemid, samuti tutvudes lähemalt Java virtuaalmasina sees töötavate lõimedega, nende ülesannete ning kasutusvõimalustega.

3.1.9 Trükkimine

Tegemist jällegi väikese teemaga, mida võiks vajadusel paigutada graafikapeatükki. Peensustesse laskumata on tegemist küllalt lihtsate vahenditega, mille abil on võimalik paljudele vajalik toiming teostada. Lihtsa trükkimisvõimaluse leidumine võrreldes sellise võimaluseta keelega muudab keele omandamise atraktiivsemaks - vähemasti olen seda mitmel korral õppurite käest kuulnud.

3.1.10 Muusika

Muusika all käsitletakse nii valmis helilõikude esitamist ning nendega oma rakenduse illustreerimist kui ka MIDI ülesehitust ning selle lihtsamaid võimalusi. Peatüki läbimisest peaks piisama tavalisemate rakendusprogrammide muusikaliseks illustreerimiseks. Rakenduse kaunistamine klippidega on piisavalt lihtne ning see ei koorma liialt põhikursuse materjale. Käskude keerukus on samas suurusjärgus nt. Multimeedia Toolbook'i omadega, mida õpetatakse edukalt ilma suurema arvutialase

ettevalmistusega inimestele. MIDI juures on läbi käidud vaid lihtsamad ja tähtsamad teemad, mille abil võib rakenduse panna helisid väljastama vastavalt kasutaja käitumisele. Veidi suuremat (kuid siiski põhikooli õppekava piiridesse jäävat) muusikalist ettevalmistust nõudvad teemad (akordid, intervallid, helistikud, saateharmoonia) on jäetud jätkukursusesse. Samuti jäi sinna kvanditud digitaalheliga seonduv. Meelepärast ja vajalikku teemat julgustas mind põhikursuse sisse panema nii Habibullini venekeelses kui Ivor Hortoni inglisekeelses raamatus ettevõetud muusikateema.

3.1.11 Graphics2D

Graphics2D teemad aitavad programmeerijat hästidisainitud rakenduste koostamisel, kuid põhimõtte poolest keerukaid ning õppija analüüsi- ning algoritmide loomise oskust arendavaid probleeme ette võetud pole. Samas on piisavalt allikmaterjali, mille peal loovate ülesannete abil vaimu teritada ning kauneid või täpseid kujutisi ja animatsioone luua. Kursuse puhul peab õpetaja hindama õpilaste loovuse- ning vastuvõtuvõimet ning vastavalt sellele kavandama tunni käigu. Silmnähtavaid võimalusi pakuvad selleks Graphics2D vahendid rohkesti, olgu siis tegemist mustrite, värviüleminekute või kujundite konstrueerimisega.

3.1.12 Swing

Tegemist suures osas samuti faktilise materjaliga. Loodut võib pigem tarvitada käsiraamatu ning näidete koguna, mille abil oma loodavaid rakendusi paindlikumaks ning kasutajasõbralikumaks muuta. Samuti peaks kirjutatu olema hüppelauaks ametliku API dokumentatsiooni juurde pöördumisel, mis muidu kipub lugemiseks liialt abstraktne olema. Kommenteeritud näidete abil on tunduvalt lihtsam mõista, mida konkreetsete käskude kirjeldamisega mõeldud ning mida analoogilistelt kirjeldatud käskudelt oodata võib. Ennistamise näite juures võiks kasutajal tekkida ülevaade, kuidas oleks süstemaatiliselt võimalik luua tagasivõetavate sammudega programme, õppides kõrvalt, millised valmisvahendid selle tarvis on juba kokku pandud. Puu abil saab kaudselt seletada, kuidas on üles ehitatud osutitel põhinevad dünaamilised andmestruktuurid. Siin on lisaks teoreetilisele seletusele võimalik silmaga vaadata, mis mille juurde kuulub ning kuidas ühendatud on. Samuti tuleb failisüsteemi graafilise puuna näitamise ülesande puhul läbi käia rekursiivse andmestruktuuriga seotud teemad ning eneselegi teadmata selgeks õppida hulk algoritme, mille eraldi omandamine võib mõnigikord tüütu, vaevanõudev ja mõttetu tunduda.

Nii Swingi kui Graphics2D seletused ja näited sobiksid kirjutiste mahtude teistsuguse paigutuse puhul pigem jätkukursuse materjalide juurde. Sel juhul sobiks sinna lisada ka mitmeid siin mainimata teemasid. Graphics2D juures tuleks kindlasti juurde juht- ja otspunktide abil joone koostamine. Samuti kujundi jagamine osadeks ning nendega omaette toimimine. Swingi puhul tuleks oma komponendi loomise kaudu läbi mängida mitmed kasutatavad mehhanismid, mis ka olemasolevates komponentides varjatuna töötavad.

3.1.13 Andmed Javas

Tegu on suurema peatükiga, kuhu on koondatud mitmed algsest keeletutvustusest välja jäänud keerulisemad, kuid tarvilikud andmete hoidmise, liigutamise ning töötlemisega seotud lõigud. Mälupuhver ning andmete vedamine on graafilise kasutajaliidesega harjunud inimesele nii enesestmõistetavad, et oleks patt vastavad olemasolevad teemad käsitlemata jätta. Massiivis sortimine on sageli vajalik. Loodud Comparator võimaldab andmeid kergesti järjestada vastavalt kasutaja soovidele ning sortimise mõistmine ja läbiproovimine lihtsustab ka tutvumist järgnevate Java Collections Frameworki võimaluste ja rakendustega.

Klassidest ja meetoditest on mul mõne muu kogumiteemalise õpikuga võrreldes antud küllaltki lühike ülevaade, kuid kogemused on näidanud, et siinses käsitluses seletatud teemad tunduvad loogilised ning võetakse küllaltki kergesti ja kiiresti omaks. Loetud materjal sobib käsiraamatuks ning teadmisi kollektsioonidest saab hiljem ikka uuesti ja uuesti üha keerulisemate rakenduste komponentidena kasutada.

Java head andmestruktuuridega ümberkäimise võimalused vääriksid lähemat eestikeelset lahtikirjutamist. Jaanus Pöial on oma loengumaterjalides kasutanud HashMap-i, siinse autori poolt on koostatud näited nimistu koostamiseks ning Dictionary tarvitamiseks. Küllalt põhjalik inglisekeelne selleteemaline käsitlus on [11], mida ka Inga Petuhhov on oma andmestruktuuride kursuses kasutanud.

3.1.14 Dokumenteerimine

Dokumenteerimine on vajalik oskus, mis õppeprogrammide juures kipub ikka ja jälle ununema ning mille puudumine hiljem iseseisvate tükkide ühendamisel kurjasti kätte maksab. Peatükis tutvustatakse Javadoci lihtsat klasside ja meetodite kommenteerimise võimalust, mille abil õnnestub kerge vaevaga täiesti loetav dokumentatsioon luua.

3.1.15 Installeerimine

Samm-sammult käiakse läbi installeerimise etapid alates Internetist sobiva faili leidmisest kuni lihtsaima omaloodud programmi abil installeerimise õnnestumise testimiseni. Kodus levinuimate Windows 95/98/Me operatsioonisüsteemides peaks toiming välja nägema üsna samamoodi. Windows NT/2000 puhul tuleb keskkonnamuutujate sättimiseks otsida Control Paneli alt Environment Variables ning seal vastav muudatus sisse viia. Et vastavad operatsioonisüsteemid leiduvad enamjaolt juba kogenuma kasutaja installeerimisalas, siis eeldatakse, et ta suudab seda kaasatuleva dokumentatsiooni abil teha. Sama lugu on Linuxi ja muude os-idega, et administraator enam eeldatavalt nii üksikasjalist juhendit ei vaja.

Lihtsa programmi kompileerimise näidis on toodud peatükki arvestusega, et oleks võimalik järjest näpuga järge ajades omale käivitamise õnnestumise ahhaa-elamus saada ning seeläbi tutvustusnäidete kaudu uue maailma saladustesse tungida.

3.1.16 Tutvustusnäited

Tutvustusnäited on kõige vastutusrikkam peatükk. Nii nagu inimese elu määrab suuresti tema lapsepõlv, nii jäävad koodi kirjutamisel alguses tekkinud tavad külge aastateks. Eriti, kui on tegemist esimese õpitava programmeerimiskeelega. Olen seda kogenud nii enese, oma õpilaste kui ka koolikaaslaste juures. Kõigepealt tööle hakanud koodi loogika ja kirjapilt sööbib alateadvusse, hilisemat on võimalik sellega vaid võrrelda. Nii otsivad algselt Basicus kirjutanud poisid kas või maa alt välja goto lause ning juhivad sellega oma programmivoolu. Samuti tundub algul staatilise main-meetodi abil kujunduse loonutele, et tegemist on ainuvõimaliku lahendusega ning hilisem konstruktori või rakendi init-meetodi selleotstarbeline kasutamine võib neile tunduda suisa vägistamisena. Siinsed tutvustusnäited on kümnekonna katse ning paarikümne eritüübilise näite seest välja valitud võimalikult vähe tupikteid sisaldav järgnevus, millele saaks igaüks toetuda oma edasiste programmide koostamisel ning mis sobib nii algajale samm-sammult liikumiseks kui edasijõudnule kiiremaks keelest ülevaate saamiseks.

Mitmetes õpikutes traditsioonilise "Hello world" asemel on siin valitud esimeseks näiteks pealkirjaga aken. Põhjuseks on Java tekstiekraanivahendite suhteline keerukus ning puisus - võrreldes näiteks selles vallas levinumate Pascali, Basicu ning ka C-ga. Mõne lühema kirjapildiga keelt näinule võib tunduda piinana, et kohe alguses tuleb luua

klass ja meetod, println-i ette System.out kirjutada. See võib saada põhjuseks Javast eemale hoidmiseks. Kui aga peaaegu samasuure vaevaga õnnestub ekraanile luua toimiv raamaken, mille pealkirja liiatigi õnnestub täiesti arusaadavalt ise määrata, siis tekitab see rahulolutunde.

Alustades tundub iga liigne sõna arusaamatu mõistena ning takistab tervikprogrammi haaramist. Seetõttu on püütud kood hoida võimalikult tillukesena. Korraga käsitletava 7 (5-9) osani ei õnnestu koodi lühendada (mis näiteks Basicu print "Tere" algajatele väga atraktiivseks teeb). Esmaste seletustega õnnestub mõned käsud ja sõnad grupeerida, nii et maksimaalselt kümnekonna minutiga suudab enamik õppureid mõista, millistest osadest programm koosneb. Tavalisest kirjutamisest erinevalt on ära jäetud isegi sõna public nii meetodi kui klassi eest, et mõistmist lihtsustada ning maagiliste tähekombinatsioonide hulka vähendada. Uues versioonis tehtavate muudatuste tõttu võib virtuaalmasin vajada ka vaikimisi nimetus paketis klassi käivitamiseks, et nii klass kui selle meetod oleks avaliku juurdepääsetavusega ning seetõttu tuleb ehk sellest lihtsustusest tulevikus loobuda. Ka on välja jäetud raami asukohta määrav setLocation - ikka lihtsustamiseks ning esimese nähtud koodi mõistmise ja käivitamise õnnestumiseks.

Teises näites on esimest muudetud küllaltki vähe, kuid arvestades, et sealjuures tuleb peale lisatavate käskluste ka näite muutmine ära õppida, peaks olema tegemist just paraja lisandusega. Nimelt tundub paljudele harjumatu, et kirjutamine ja kompileerimine käivad sootuks erinevate programmide abil. Samuti, et kompileerimine pole mitte ühekordne tegevus, vaid tuleb ette võtta igal korral pärast koodi muutmist. Eraldi tekstiredaktori ning kompilaatori tarvitamine on hea ning soosib mõistmist, et (Java) programmifailid ei ole ühe kindla arenduskeskkonna pärusmaa, vaid kompilaator on eraldi programm ning sinna ette antavad tekstid on tavalised tekstifailid, mille loomisega peaks igasugune tekstiredaktor hakkama saama.

Järgnev liikumise näide pakub konkreetsete operatsioonide staadiumis õpilastele tähtsa algmaterjali, mis võimaldab mõnikord mitme tunni vältel huvitavate ülesannetega tegelda. Teistelgi on tegemist küllalt köitva ning kaasakiskuva, kuid samas lihtsasti ja kiiresti mõistetava toiminguga. Kuna küllalt ruttu tekib mõnele idee oma töövaeva vähendamiseks, siis õnnestub enamasti selle koha peal tsükkel kergesti selgeks saada. Samuti selgineb muutuja mõte ning katsetamiste tulemustena ilmneb, mis erinevus on üksteisele järgnevatel või üksteise sees olevatel tsüklitel. Sinna juurde paigutatud

värvimäng pakub noorematele tutvujatele palju. Teoreetilisemat laadi kursuses on jäänud see näide pigem kiirematele lahendajatele ning suurem osa seltskonnast sellega ei tegele.

Joonistamise puhul on osutunud kasulikuks vähemasti esialgu teha kogu joonistustöö alamklassi paint-meetodi sees, olgugi, et see ei pruugi olla lühim lähenemine. Nii soovitab ametlik dokumentatsioon ning nii tekib vähem seletamatuid probleeme. Samuti olen avastanud, et joonistusoskusega akna ning selle käivitamise käsud on olnud kasulik paigutada eraldi klassidesse, sest sel juhul tuleb selgemalt välja klassist eemal tehtud objekti kasutuselevõtt. Alles järgmise näite juures pannakse need teemad ühte klassi kokku. Edaspidi on loodud konteinerisse paigutatav komponent (rakend) ning see toodud nähtavale loodava raami sees. Sellist lähenemist on edasisel õppimisel soositudki, kuid teiste sarnaste õpetuste ning näiva lühiduse tõttu on raami alamklassi koostamine esiotsa sisse jäetud. Tõenäoliselt tuleb pärast mõningaid katseid loobuda raami alamklassist selles staadiumis, sest kuna on tegemist esimese Javakeelse graafilise programmi kirjutamise malliga, siis kipub see mällu jääma ning edaspidi on küllalt raske harjuda konteineri sisse paigutatavaid /uusi, muid/ komponente looma. Samuti võib konkreetseid operatsioone arvestavas kursuses kaheks klassiks jagamise etapi vahele jätta. Sel juhul teoreetilist arusaamist enamikul alguses nagunii ei teki ning kord mitme ning siis taas ühe failiga toimetamine lisab pigem segadust kui selgust. Veebilehel oleva rakendi etapi võib vastavalt kasutajate omandamiskiirusele siit välja jätta. Aeglasema taipamisvõimega grupi puhul sobib rakendi juurde pöördumine pigem tutvumisnäidete lõpus - siis kui keelega on jõutud enam harjuda.

Kasutaja andmete järgi ringide joonistamise näide võiks jääda mitme tunni aluseks. Vaid päris kogenud programmeerijate puhul võetakse see teadmiseks ning suudetakse teiste keelte pealt saadud analoogiate põhjal edasi minna. Nüüd, kus lihtsamad arvutamise ning joonistamise oskused olemas, võib nende põhjal küllalt keerukaid ja arendavaid jooniseid koostada, ilma et otseseid arvutikäske peaks oluliselt juurde õppima. Nii võib harjuda mitmesuguste algoritmide koostamisega. Keerulisemate jooniste puhul võib tulla rohkem sisendandmeid ning sellega seoses ka tekstivälju. Funktsiooni graafikute joonistamisel võib kasuks tulla loodud alamprogramm. Mõne aja pärast hakkavad mitmedki tundma, et täiesti kasutuskõlblike programmide koostamine võib juba jõukohane olla.

Ka hiire teadete püüdmise juures on esitatud võimalikult lihtne ning samas kergesti edasi arendatav näide. Kuularite seletamise jaoks pole sisemise klassiga näide

hea, kuid kuna lähem analüüs on nagunii teooriaosa sisse jäetud, siis peaks tutvustamise jaoks just siinne olema sobiv. Tekstiekraani näide on pandud muudest eraldatuna lõppu. Iseenesest aga sellise lihtsa sisend-väljundnäite ning eelnevate katsetuste tulemusena peaks olema võimalik ka käsureal märkimisväärne osa vajaminevatest lihtsamatest abiprogrammidest kokku panna. Kel aga tahtmine tõsisemate rakenduste koostamise juurde asuda, see peaks siirduma raamatu algusse ning süvendatumalt õppima hakkama.

3.2 Java jätkukursuse konspekt

Java jätkukursuse konspekt (Lisa 2) on õppevahendiks ja abimaterjaliks kogenenumatele programmeerijatele. Ei pretendeerita ülevaate andmisele kõigele Javaga seonduvale. Mitmetel teemadel on lugeja püütud aidata tasemeni, kus ta suudab keskmise raskusastmega programme ise kokku panna ning edasi juba mõista dokumentatsiooni ja keerulisemaid õppematerjale. Jätkukursuse teemad on suhteliselt iseseisvad ning üldjuhul ei eelda järgnev teema eelnevaga tutvumist. Seetõttu saab otsida sisukorrast sobiva peatüki ning see omandada, soovitavalt näiteid arvutil käivitades ning oma fantaasia järgi edasi arendades. Küll aga ei seletata enamasti üle põhikursuses käsitletud mõisteid ja teemasid ning kui taust tundub segaseks jääma, siis tasuks uurida nii põhikursuse konspekti kui püüda omale mujalt abimaterjale otsida.

Raamatu esimene pool on pühendatud mitmesugustele joonistamise ja graafikaga seotud teemadele. Lisapakettidest on tutvutud vaid Java3D-ga. JMF (Java Media Framework) ning JAI (Java Advanced Imaging) on jäänud puutumata. Sellegipoolest peaks siinse õpetuse ning ametliku dokumentatsiooni najal olema võimalik lahendada enamik graafikavallas ettetulevatest probleemidest. Graafikaga seotud arvutuste põhjalikumaid õpetusi võib leida raamatust[40].

Kirjutise teises pooles on läbi vaadatud võrgu- ja andmebaasiprogrammide koostamine ja nendega seonduvad teemad ning seal on graafikat kasutatud minimaalselt. Autoril on mitmegi sinna ossa sobiva peatüki jaoks juba näidete kogu koostatud või lihtsalt vastav teema kirjutamisvalmina meeles mõlkumas. RMI kõrvale sobiks universaalsem CORBA koos seletustega, kuidas Javakeelseid programme muukeelsetega ühendada. Laiemat tutvustamist vajaksid J2EE spetsifikatsiooni põhjal loodud vahendid, samuti arutlus, millal on neid mõistlik ja millal mõttetu kasutada. Ka laieneva kasutusega XML-i teemad kuuluksid raamatu siia ossa. Kuna vastavad vahendid on võetud JDK 1.4

standardse API koosseisu, siis see justkui soosiks nende kasutuselevõttu. Samas staatuses on ka JCE ehk krüptograafiavahendid, kus nii DES-i, RSA kui muude algoritmide abil andmed salastada ning sobiva võtme leidumisel taas lugeda.

Kirjutise lõpetab muusikaosa, mille mõistmiseks ja edukaks rakendamiseks tuleks meelde tuletada kooliajal nii füüsika- kui laulutundides õpitu.

3.2.1 Java3D

Kolmemõõtmelise graafika peatükis on ette võetud Java3D standardne lisapakett ning tutvustatud nii sealseid lihtsamaid kui veidi keerulisemaid kujundeid ja põhimõtteid näidete varal. Lõpuks on selle abil kokku pandud kosmoses liikumist imiteeriva keerulisema programmi süda, mille peale on juba igaühel võimalik oma oludele sobilik rakendus ehitada.

3.2.2 Rekursiivne joonistamine

Rekursiivse joonistamise juures tutvustatakse iseennast korduvalt välja kutsuva alamprogrammi rakendamist jooniste loomisel ning reaalsust imiteerivate programmide koostamisel. Sellest peatükist võiksid kasu saada nii graafilise kujunduse koostajad kui ka inimesed, kellele rekursioon huvitava ja vajalikuna tundub, kuid tavaline andmepõhine lähenemine on kuiv või raskesti arusaadav tundunud.

3.2.3 Maatriksarvutused

Maatriksarvutuste peatükis võetakse läbi tasandil ja ruumis nihutamise ja keeramise puhul kasutatavad tehted. Näidatakse, kuidas suure punktide arvuga kujundi korral võib masina arvutusvaeva oluliselt vähendada ning seeläbi programmi tööd kiirendada. Nõuab veidi enam matemaatilist pühendumist, kuid vähemasti alates keskkoolist peaks igaühele tõsisema süvenemise korral arusaadav olema.

3.2.4 Joonistamine

Joonistamise peatükis tuletatakse meelde Java graafikakomponentide tööpõhimõte ning joonistamisel vajalikud käsud. Edasi vaadeldakse ükshaaval Java teekide poolt pakutavaid filtreid ning õpetatakse neid valima ja kasutamisel omadusi määrama. Peatükis õpitu peaks sobima nii fotode väikeseks iluraviks kui uute piltide kombineerimiseks ja sünteesiks.

3.2.5 Pilditöötlus

Edasi minnakse pilditöötlusse veel sügavamale ning õpetatakse üksiku pikseli kaudu andmeid küsima ja määrama. Ehkki näited on koostatud lainetuse loomise kohta, saab siin omandatud oskusi kasutada nii pildiandmete analüüsi (nt. kirjatuvastus), arvutitevahelise ülekande kui mitmesuguste animatsioonide loomise juures. Koos ekraanipildi kopeerimisega võimaldavad siin peatükis omandatud oskused koostada rakenduse, mis analüüsib mõne teise programmi tegevust ning vajadusel suunab või testib seda.

3.2.6 Turvalisus

Turvalisuse alla on kokku koondatud mitmed Java standardvahendite selleteemalised osad. Avaliku võtme krüptograafiaga seotud toimingud on sarnased nii kaasatulevate vahendite abil arhiivi signeerimise kui ka otseselt baidijadale digitaalallkirja andmise puhul. Samuti vaadeldakse peatükis, kuidas kindel olla, et käivitatav programm ei saaks ei kohalikule masinale kurja teha ega ka võrgu kaudu minna ebasoovitud toiminguid sooritama.

3.2.7 Andmebaas

Andmebaasi juures näidatakse kommenteeritud piltkirjas, kuidas luua lihtsaim ühendus MySQL-andmebaasimootoriga ning selle kaudu andmeid vahetada. Samuti õpetatakse servleti abil andmebaasile lihtsat veebiliidest koostama. JSP puhul jätkatakse veebiserveri programmeerimist ning alustades lihtsatest paarirealistest lehtedest jõutakse andmete hoidmiseks ning arvutustöödeks mõeldud oani.

3.2.8 Lühiteemad

RMI-st on toodud üks terviklik töötav näide, mille põhjal on võimalik vajadusel tunduvalt suuremate oskustega kaugkasutatavaid objekte luua. Tõlkimise, atribuutide ning numbriformaadi juures piirdutakse samuti lühidalt, sest informaatika seisukohalt on teemad lihtsad. Kui rakenduste koostamisel vastavad vajadused tekivad, siis peaks siin toodud näidete põhjal analooge konstrueerides suutma igaüks enamiku oma soovidest rahuldada.

Erindite puhul on tegemist põhikursuses alustatud teema jätkuga. Kui seal piirduti olemasolevate erindite püüdmise ning programmide koostamiseks hädatarviliku lihtsama töötlusega, siis siin luuakse ja lastakse lendu ka ise erindeid ning tutvutakse vähemlevinud reageerimisvõimalustega.

3.2.9 Muusika

Muusika puhul on samuti tegemist põhikursuse materjalide jätkuga. Siia on jäetud pikemat mõtlemist vajavad teemad. MIDI saateautomaat võiks huvi pakkuda ka muusikaga tõsisemalt tegelevale inimesele. Seal on tutvustatud nii etteantud duuride järgi taustaakordide koostamist kui ka igal korral uuena kõlavat rahvaliku viiuli saatepartii sünteesi. Kvanditud heli mõistmine võib algul veidi aega nõuda, kuid põhisisu tabamisel ei tohiks sealt soovitud suunas edasiminek enam liialt raske olla. Õppetüki edukalt omandanu peaks suutma koostada heliredaktoreid ja luua heli salvestavaid ning lihtsamal moel analüüsivaid rakendusi.

3.3 Javaskripti konspekt

Konspekt (Lisa 3) on mõeldud kasutamiseks põhiliselt abimaterjalina loengute ja praktikumide juures, kuid sellest peaks kasu olema ka iseõppijatele. Terviklikku ülevaadet pole taotletud, sest eesti keeles on ilmunud mitmeid selleteemalisi raamatuid [39] [61] ning veebilehti [8]. Pigem on püütud anda ülevaade teemadest, mis on seni puutumata jäänud või millest kergelt üle libisetud.

Alustuses on lühidalt tutvutud peamiste tarvilike käskluste ning konstruktsioonidega. Varasema programmeerimiskogemuse või kõrvaliste seletuste abil peaks nende kaudu omandama oskuse lihtsamate programmide kirjutamiseks, vajadusel näidetest analooge abiks otsides.

3.3.1 Pildid

Piltidele on pühendatud vaid paar lehekülge, kuid sealt saadud tehniliste oskuste abil ning ülesannetest või oma peast ideid otsides peaks saama kokku panna päris mitmekülgseid graafikalahendusi nii mõne teema tutvustamiseks, protsessi jäljendavaks interaktiivseks mudeliks või lihtsalt silmailuks. Rohkem graafikaga tegeleva kursuse puhul tasuks punktpiltide põhjal panna kokku nii näitlikustavaid jooniseid, animatsioone kui simuleerivaid mudeleid ja mänge. Põhikooli arvutiringis pusimiseks piisab vaid piltide teemast mitmeks kuuks.

3.3.2 Objektid

Sisseehitatud objektide peatükis on seiluri standardvahendite ülevaade. Sealt peaks kasutaja avastama enese jaoks skaala, mida on põhjust asuda ise kirjutama. Võrreldes mõne muu keelega (näiteks Javaga), on kaasatulevate oskuste kogum suhteliselt väike, kuid kuna nii käsklused kui objektid on küllalt paindlikud, siis saab nende abil sooritada

enamiku vajaminevatest toimingutest. Pärimist ning objektorienteerituse muid nüansse ei pruugi väiksemates Javaskripti programmides kuigivõrd vaja minna, kuid siinne kokkupuude loob kogemuse, et pärimine ei pruugi sugugi alati vaid klassidega seotud olla. Isenditega tegelev programm pakub õpetajale võimaluse tutvustada objektorienteeritud andmebaaside tööpõhimõtteid ning nendele esitatavaid nõudmisi. Lähemalt tutvutakse massiiviga ning selle abil lehekülje muutujates asuvate kasutaja silma eest peidetud andmete väljastamisega päringu peale. Isendite juures käiakse süstemaatiliselt läbi Javaskripti objektide loomise ja kasutamisega seotud teemad. Õpitu peaks aitama nii mõista olemasolevate objektide toimimist kui ka aidata enesele sobivaid objekte koostada. Samuti võiks peatükk laiendada edasijõudnute objektorienteerituse alast silmaringi, näidates nii Javast kui C-st põhimõtteliselt erinevat klassideta pärimist.

3.3.3 Regulaaravaldised

Lähemalt on üle vaadatud regulaaravaldised, mis teistes samateemalistes õpikutes kipuvad jääma tähelepanuta, kuid samas on küllalt tõhusaks abivahendiks kasutajalt saadud tekstide analüüsil. Need lihtsustavad märgatavalt vormist serverile saadetavate andmete mustrile vastavuse kontrollimist (näiteks, kas kasutaja on sisestanud korrektse kuupäeva). Arvestades, et versioonist 1.4 on ka JDK standardpaketti lisatud regulaaravaldised, aitab nendega siin tutvumine kergendada edaspidist sõnetöötlust.

3.3.4 Kihid

Keerulisema ning kasutajast sõltuva kujunduse puhul ei saa üle ega ümber kihtidest. Need on hädavajalikud nii menüüde loomisel kui mittetekstiliste animatsioonide koostamisel. Ühes kihtidega on näidatud ka hiire teadete püüdmine ning selle realisatsioon nii Netscape Navigatori kui Internet Exploreri puhul. Ka kihtide juures tuleb vahetevahel seilurite erinevust arvestada.

3.3.5 Andmed

Lõpuossa on koondatud mitmed andmetega seotud toimingud. Küpsised on hädatarvilikud, kui tahetakse serveriprogramme kirjutamata talletada väärtusi kasutaja järgmise sisenemise tarbeks. Andmete lugemine aadressireal võimaldab Javaskripti abil kokku panna taolisi teenusehekülgi, mis muul juhul jäävad vaid serveriprogrammide pärusmaaks.

3.3.6 Edasiarendusvõimalused

Konspektist on puudu veel mitmed teemad, mis terviklikkuse mõttes sinna võiksid kuuluda ning mida eesti keeles pikemalt lahti seletatud ei ole. Küllalt tähtis oleks raamide ja akendega seonduv, mis võimaldaks sobival ajal vajalikke andmeid aknas näidata ning aknaid paigutada ja sulgeda. Ka vormi elementide süstemaatilisem ülevaade oleks vajalik, et kasutaja teaks, mida millestki oodata võib ja kuidas omadusi määrata. Samuti kuluks käsitleda serveris programmeeritud veebilehtede ning Javaskripti koostööd, et õnnestuks võimalikult optimaalselt veebilehestik koostada ning kõik kliendi juures tehtavad tööd ilma aeganõudvalt uusi lehti avamata sooritada.

3.4 Veebiprogrammeerimise ülesannete kogu

Koostatud ülesanded (Lisa 5) on kavandatud abiks nii õpetajale kui iseõppijale, pakkudes mitmesuguseid lahendust nõudvaid probleeme ning andes mitme taseme tõttu ühe võimaliku tee lahenduse poole liikumiseks. Kuna korduvalt on autor kuulnud õpetajate nurinat väheste kokkupandud arvutialaste ülesannete üle, siis siinne kogu võiks seda lünka veidi täita. Kogu pole sugugi ammendav ning näiteks andmebaaside teemal sobiks hulga põhjalikum komplekt lahendamiseks, kuid nii algajad kui ka märkimisväärne osa edasijõudnutest peaksid leidma omale sobivaid edasiarendamist vääriva näiteid. Osa ülesannetest võib sobida mitmel pool otse kasutada, kuid soovitav on võtta leitud idee aluseks ning vastavalt õpitud taustale lahendusteed lihtsustada või sellele nüansse juurde mõelda. Ideed on enamasti küllalt universaalsed ning sobivalt kohandades peaksid lahendamiseks sobima mitmesuguse ettevalmistuse korral. Mitmelgi puhul on pakutud enam võimalusi kui lihtsamaks oskuste harjutamiseks tarvilik. Kui ülesanne liialt keerulisena tundub, siis ei pea seda mitte kõrvale jätma, vaid tuleb vaadata, kas õnnestub mõne punkti eemaldamise või lahtiseletamise abil probleem õppijatele suupäraseks teha. Enamik siin toodud ülesandeid pole seotud konkreetse keelega, vaid samu lahendusi on võimalik saavutada keelest sõltumata.

Märgatav osa ülesandeid on jaotatud kolmeks alapunktiks, kus iga järgmine peaks eelmisest paar-kolm korda keerulisem olema. Tegevused on püütud jagada nõnda, et nii esimene punkt eraldi kui kaks alguspunkti koos annaksid terviku ning sinna pidama jäädes ei tekiks poolelijäämise tunnet. Koostamise ajal on püütud nii arvestada, et ülikooli 90-minutilise praktikumipaari ajal jõuaksid kõige tugevamad lahendada kogu kolmest punktist koosneva ploki. Enamik saaksid selle ajaga hakkama kahe astmega ja mõned

neist jõuaksid raskeima taseme läbi mõelda. Ning ka kõige nõrgemad või lihtsalt õpilased, kel sel korral pea lahti pole, peaksid vähemasti esimese tasemega hakkama saama. Ülesanded, kus on rohkem alapunkte, on mõeldud kas algõppeks või siis väga erisuguse tasemega õppijate rühma tarbeks. Esimesel juhul on teemad võimalikult väikesteks osadeks jaotatud, et igaühel õnnestuks tunni jooksul vähemalt üks või kaks eduelamust saada. Teisel juhul on arvestatud, et tugevamad võivad mõne alumise taseme läbi katsetamata jätta (eeldusel, et nad mõtetes sellest üle on) ning siis raskemate etappide kallale asuda.

4 Näidete koostamise ja esitamise põhimõtted.

Näidete hulk, pikkus ning osakaal programmeerimisõpikutes kõiguvad suurel määral alates täisteoreetilisest jutust ning tegelike rakenduste ja kasutusvõimaluste vihjete täielikust puudumisest kuni üksikute kommentaaridega väljatrükitud lähtekoodi või toimuvate sündmuste üksikasjaliku kirjelduseni [38]. Sõltuvalt inimtüübist ning ainevaldkonnast võib kasu olla nii mõlemast äärmusest kui tasakaalustatud keskosast. Võib isegi juhtuda, et lihtsam on lugeda teooriat ühest ning katseid teisest kohast kui et püüda nende ühitatud kujust selget pilti saada.

Puhas teooria on enamasti kirjutatud küllaltki loogiliselt ning kokkuvõtlikult, nii et sealt on võimalik näpuga järge ajades kõigepealt üldpõhimõtted selgeks saada ning siis tasapisi sügavamale minema hakata. Nõnda võib küll otseste probleemide lahendusteni jõudmiseks hulga aega kuluda, kuid samas - olles kord ülalt lähenedes alla välja jõudnud - saab sarnaseid olukordi eelpool kogetud teadmiste abil juba hõlpsamini lahendada. Kui üldpilt selge, on ka kergem otsustada, kas üksikjuhtudele lahendusi leidub või mitte. Lähenemise puuduseks on aga, et kui otsese ülesande tarvis on vaja läbi käia mitmeid valdkondi, siis kulub palju aega üldpõhimõtete omandamise peale ning mõni suhteliselt väike osa võib jääda komistuskiviks edasise töö jätkamisel. Ka võib mõnele inimesele üldistuste tegemine ja nende järgimine osutuda tunduvalt raskemaks kui otseste näidete omale vajalikuks kohendamine.

Praegu kiirelt arenevas arvutivaldkonnas mitmedki üldisena näivad arusaamad ei pruugi mõne aasta pärast enam samasugused olla. Sarnaste olukordade lahendamiseks võib leiduda tunduvalt paremaid võimalusi ning praegused tavad võivad pakkuda vaid ajaloolist huvi ning kogemusi uute vahendite loomisel, mitte aga ainuvõimalikku lahendusteed.

Töötavast näitest alustades võib alati kindel olla, et vähemalt etteantud kitsendavate tingimuste korral lahendus toimib. Seda polegi nii vähe. Kaugelt ja kõrgelt lähenedes tuleb mõnikord enne kurja vaeva näha kui kasvõi midagi pisikest tööle hakkab. Kui aina proovid ning püüad kõike üha lihtsamaks teha, aga ikka ei õnnestu, mõjub see halvasti ning tekitab tahtmise kogu loole käega lüüa. Näidet muutes saab pidevalt kontrollida, millisest hetkest alates enam tulemus ei vasta oodatule. Nii on võimalik pisiosi muutes süsteemi tundma õppida ning omandatud põhimõtete alusel edaspidi

tarvilikke tulemusi saada. Puuduseks jääb muidugi, et tehtud lihtsustused võivad jäävad liiga suurteks ning omandatud teadmisi pole võimalik muutunud keskkonnas kasutada.

Kui kahte lähenemist võrrelda, siis need sarnaneksid tee otsimisele võõras linnas. Teoreetilise lähenemise puhul tuleks ette võtta linna üldplaan, selgitada jõe, raudteede ning suuremate tänavate paigutus ning alles seejärel asuda aadressi järele õiget paika otsima. Näidetele aga ehk vastaks kergelt ekslev teeleasumine oletatava sihtkoha lähedasest paigast teadmisega, et lähtekohta võib alati tagasi minna ning sealt saab uuesti teises suunas otsinguid alata. Võib tunduda, et esimene viis on palju parem, kuid salapärases paigas hoovi peal asuva kingsepa leidmiseks võib ekslemisest hulga enam kasu olla kui ebatäpse kaardi uurimisest.

4.1 Lühikesed näited

Näite juures pean tähtsaks, et see oleks piisavalt lühike ning haaratav, kuid samas terviklik ja töötav. Sellise lähenemisega õppijaid on mitmeid, kuid sellegipoolest kipub enamik arvutiõpikute koostajaid näiteid kas tervikluse hinnaga lühendama - lootes, et õppurid kerge vaevaga puuduvad osad juurde mõtlevad ja tulemuse tööle saavad - või ilu- ja viisakusvahendeid lisades tahtmatult pikendama, raskendades tunduvalt võimalust näidet korraga haarata. Raamatu autorile iseenesestmõistetavad käivitamiseks tarvilikud read võivad õppija jaoks osutuda pea ületamatuks raskuseks, kui talle pole veel tekkinud harjumust ja korrapära, millised sõnad millises järjekorras millist sisu tähendavad. Kuigi võib tegemist olla pealtnäha lihtsate asjadega, suudab inimene siiski omandada parimal juhul paar-kolm uut mõistet tunnis ning kahju kui sisuni jõudmiseks tuleb enne ekselda tehniliste murede rägastikus ning abi nende lahendamiseks ei paista ka kusagilt tulema.

Õpetajal tekib paratamatult kiusatus näited lisaks heale sisule panna ka välja nägema. On ju võrratult ilusam näidata ekraanil korralikult joonistatud ning pealkirjastatud ja mõõtmisandmetega varustatud kuupi kui lihtsalt mingit ähmast udukogu, eriti kui selline tulemus on võimalik saavutada vähem kui minuti klõbistamisega. Õppijale aga on sisu ning kujundamise osad programmis ühtviisi udused ning vajaliku koha üles leidmise aeg võib kasvada märgatavalt. Inimene suudab korraga meeles pidada ja analüüsida viit kuni seitset (erijuhtudel 3-9) tema jaoks eraldatavat tervikut; ülejäänud andmed tuleb paigutada aeglasemalt töötavasse püsimällu ning sealt kätte saamiseks tuleb nad uuesti meelde tuletada. Ka ise olen näidete koostamisel läinud mõnikord kaunistuste ja korra teed, kuid enamasti olen märganud selle tee mõttetust.

Õppijate hulgas leidub ikka keegi, kellele just need paar-kolm kaunistamise tarvis loodud lauset komistuskiviks saavad ning keda sealt üle aidata tuleb. Paremal juhul õnnestub neil siis sisust aru saada, kui kõik üleliigsed (sisusse mitte puutuvad) laused maha kustutada ning jätta alles vaid kriitiline mass, mille puhul rakendus veel oma loodud eesmärgile vastab. Kui sellest surnud punktist on üle saadud, siis võib ta ise juba rahumeeli sinna kaunistusi lisada, saades kõigest toimuvast aru. Küll aga tuleb mõnikord sootuks paralleelne näide kirjutada ja kasutajaliides otstarbekaks ning meeldivaks muuta, et oleks midagi köitvat tahvlile näidata ning ka õppijal tekiks korralikuma töö nägemise harjumus.

4.2 Näidete seeriad

Kui näited tehakse samast tüvest, siis peaks olema kergem ka ühelt järgmisele edasi minna. Enamasti piisab tähtsamate vahendite katmiseks kahest-kolmest näitest, kuid mõnikord on ka viiel korral sama teed järjest edasi arendatud. Kord ühes meeldivas ning ladusalt loetavas väljamaa õpikus oli sama näidet edasi kasvatatud kaheksa korda. [5] Paberi peal kulutab selline trükkimine ohtralt ruumi, eriti kui pikki lõike korduvalt trükkides seal vaid üksikuid ridu muuta. Arvuti puhul mure väheneb, sest tekst võtab andmekandjal suhteliselt vähe ruumi ning mõnekümnerealisi näiteid mituteist korda sarnaselt trükkida pole kuigi kulukas. Kui näide panna viitena avanema kõrvalaknas, siis pääseb sinna kergesti hiirevajutuse abil ning teksti üldine kude jääb lõhkumata. Paljude lisavõimaluste puhul on mõistlik mitte neid ükshaaval olemasolevale juurde lisada, vaid lihtsa põhja peal üksikult esitada ning lõpuks kõik kokku panna. Kui lõigud on lihtsad, siis pärastine tuttavate sõltumatute käskude kokkupanek pole enam raske.

4.3 Kommentaarid

Näidete koostamisel tundub ahvatlev lüüa kaks kärbest ühe hoobiga, lisades lähtekoodi sisse pikad kommentaarid algoritmi toimimise ja tehniliste vahendite kohta. Nii aga koodi kompaktsus kaob ning pikki tehnilisi ja autoriõiguslikke seletusi lugedes on algajal küllalt raske jõuda tuumani, kuidas toimib programm. Koodis olevad põhjalikud kommentaarid on otstarbekad koodi dokumenteerimise juures, kui programmeerijal on mugav kõik ühes kohas kirja panna ning lasta hiljem javadoc-il või muul sarnasel programmil dokumentatsioon loetaval kujul vormistada.

4.4 Näidete valimine

Suurema näidete hulga puhul läheb lahendamise aluseks sobiva algmaterjali leidmine küllaltki keeruliseks. Kui õnnestuks ettevõetavad teemad ühte loogilisse ritta

paigutada, ka siis ei õnnestu enamasti teema igale osale näidet koostada ilma, et see muudest näidetest märgatavat osa ei kordaks. Et teemad kipuvad omavahel samuti läbi põimuma, siis see lubab ka näidetel osaliselt kattuda, teisalt aga peaaegu et kaotab lootuse teha iga teemalõigu kohta ilus ümmargune terviklik lühike arusaadav piiritletud näide, mida ka kerge lugeda oleks.

Õpikutes on olukord lahendatud mitmeti. Kord on toodud lehekülgedepikkune tervliklik näide ning seejärel asutud keerulisemaid ja eeldatavasti raskemini mõistetavaid ning teema jaoks tarvilikke kohti selgitama. Selline lähenemine sobib edasijõudnutele tuttava teema seletamisel, kus lugeja on kohe võimeline üldpilti haarama ning ülesehitusse soovi korral kas või ise osi lisama või sealt eemaldama. Sellisele lugejale piisab mõnikord suisa paljast üksikute abistavate kommentaaridega lähtekoodist. Edasised seletused võiks ta vajaduse korral ning kontrollivõimaluse olemasolul ka ise valmis kirjutada. Nende läbilugemine võib koodi uurimise käigus tekkinud arvamusi kinnitada ja üldistada või valearvestusi avastada. Vastasel juhul ilmenvad need alles masinal katsetamisel.

Eelmises lõigus kõne all olnud tekst sobib ka teemaga vähem tuttava, kuid muidu usina ning lahtise peaga õppuri tarvis. Tema saab esialgsest koodi uurimisest parimal juhul uduse üldpildi või lihtsalt aimu koodi ligikaudse pikkuse ning väljanägemise kohta. Masinas rakenduse nägemine ja kasutamine veenab, et tegu pole mitte lihtsalt kokku kirjutatud ridadega, vaid et masin nende põhjal soovitud töid ette võtta mõistab. Teemat vähem tundva usina kasutaja tarvis on korralikud seletused määrava tähtsusega. Need avavad raskemad kohad, millest pole olemasolevate teadmiste abil ilma lisamaterjalideta võimalik aru saada, samuti seovad näited eelnevalt teatuga. Kasuks tulevad teadmisi kinnistada aitavad kokkuvõtvad laused. Kui ta nüüd uuesti esialgse tervikliku lähtekoodi läbi mõtleb, siis peaks mõistmine olema ligikaudu samal tasemel kui esimest korda koodi lugeval teemaga tuttaval inimesel.

4.5 Näidete kättesaadavus

Kui näited lähevad rohkem kui lehekülje pikkuseks, siis tavatsetakse need mõnikord lisadena lõppu kirjutada. Arvan siiski, et kui näidet kasutatakse vaid ühe peatüki piires ning mujal sellega enam ei tegelda, siis tuleks näide teksti vahele kirjutada, nii et see lugedes silma alla jääks. Vaid harva viitsitakse raamatu lõppu lapata ning keskelt seletusi lugedes lõpus näpuga järge ajada. Paar lehekülge edasi-tagasi vaadata on tunduvalt lihtsam. Ehkki paberile trükitud kujul on koodi mugavam lugeda kui

arvutiekraanilt, tundub mõnikord mõistlikum näidete lisadesse kirjutamise asemel need kettaga kaasa anda, et kasutaja saaks neid siis lugeda, käivitada, veenduda töötamises, näidete võimalusi muutmise teel kontrollida ning mujale sobivaid lõike kopeerida. Paberile trükitud näited on asendamatud rahulikult suvel metsas kuuse all istudes keele võimalustesse ja ehitusse süvenemiseks, ent küllalt sageli on just "elavatest" näidetest kasu.

Sama näite mitmekordsel kasutamisel tuleks tähtis lõik ikka uuesti jutu juurde kopeerida, olgu või lühikeselt ja väikeses kirjas. Kui kord on juba terviklik kood üle vaadatud, siis muutub väljatoodud osade sobivasse paika asetamine lihtsamaks, seletuste juures töötava teksti silmitsemine aitab hoida kontekstist liigselt kaugenemast ning vältida mõtete uitamist muremaale, kus tekib pelgus, mida on võimalik täide saata ning mida mitte. Kui eelnevalt töötavana nähtud käsud on silme ees, siis jääb küllalt suur tõenäosus, et tekstis sinise värvi asendamisel rohelisega muutub ka ekraanil sinine roheliseks ning täiendavaid probleeme muutusega koos ei teki. Kirjeldusega kaasneb ikka oht - eriti kui lugeja materjalist tugevalt üle ei ole - et autor jätab mõne segase asjaolu mainimata või läheb tal midagi kõrvust mööda ning pealtnäha lihtne täiendus ei lase kogu süsteemil enam uuesti tööle hakata ning peab taas hakkab algusest uurima, et mis on mille jaoks ning kus viga tekib.

4.6 Näiteks lõik töötavat koodi

Kasutajale näidatav programm või teksti kopeeritud väljund peab olema sama näide, mitte selle ilustatud koopia. Töötava näitega kaasneb peale paari tutvustatava koodirea veel palju muudki: kogu käivitamiseks ja kasutamiseks vajalik taust. Olen ise kurja vaeva näinud paarirealise servleti tööle saamisega, sest näide oli tehtud küllalt lihtne, selle vaatamiseks esitatud koopia aga sisaldas eneses veel vahendid teistele lehtedele siirdumiseks ja abiinfo tarvis; need aga muutsid väljundis näpuga järje ajamise võimatuks. Rida-realt algvariandi ning oma töö kontrollimise võimalus on väga tähtis uute vahenditega tutvumise algjärgus. Nii õnnestub erinevused ning võimalikud vea põhjused avastada ning jõuda suhteliselt kiiresti sobivalt töötava koodini, millesse muutusi viies saab seda tasapisi põhjalikumalt tundma õppida. Räägitakse, et väikestel lastel pidi jäljendamine olema kõige levinum ning tulemuslikum õppimise moodus, kuid arvan, et pea sama kehtib ka mujal uude maailma sisenemise puhul, milleks programmeerimisega alustamine ju on. Kui koodi sisse pandud ilustused muudavad selle lihtsalt kohmakamaks

ning raskemini mõistetavaks, kuid otseselt ei välista veel rida-realt sisu jälgimist, siis samaeesmärgilise kuid teiste käskudega koostatud programmi juurest on juba küllalt lootusetu jälgida, mis millest kuhu tekkis.

Seletava teksti vahele pikitud koodinäited peavad olema sinna töötavast programmist kopeeritud, mitte niisama sisse kirjutatud. Kuigi võib tunduda, et lihtsate käskude kirjutamisel pole võimalik vigu teha, õnnestub siiski pea igas kümnendas näitelõigus unustada sisse mõni tarbetu rida või kusagil täht valesti kirjutada, nii et näite mõte hägustub ning kasutaja ei saa seda enam pidada ilusaks etaloniks, millesse uskuda ning mille järgi oma tegemisi sättida.

Nii õpikus kasutatud kui muul moel õppijat abistavad näited olen püüdnud koondada kokku, et neist koodi kirjutamisel abi võiks leida. Algselt koostatud kümnekonna põhinäite abil saab niidiotsa kätte suurema osa tavavahendeid sisaldavate programmide koostamiseks. Nende seast sobiva leidmisel ei teki loendi olemasolul kuigi suuri raskusi. paari minutiga õnnestub järele vaadata, kas valmiva töö loomiseks on abi loota või mitte.

4.7 Näidete paigutamine veebis

Õppematerjali koostamisel kaetud teemade arv aegamööda kasvas, samuti oli otstarbekas mõne tähtsama lõigu mitmest küljest iseloomustamiseks teha rohkem kui üks näide, sest mõnele õppijale sobivad lühikesed ja kuivad, mõnele lihtsamakoelised kuid pikemad näited. Nõnda kasvas näidete arv tuhandeni. Sellises hulgas korralikult orienteeruda ei õnnestu, tuli hakata koostama muid vahendeid, et otsimise aeg näitest saadavast kasust suurt osa ära ei kulutaks. Õppetöös andis hea tulemuse näidete jaotamine tundide järgi. Uue teema algmaterjaliks vajalikud tunni näited moodustasid ühe kataloogi. Enamiku vahendeid sai samast kätte. Kui aine koosnes mõneteistkümnest praktikumipaarist ning kümnekonnast eraldiseisvast loengust, siis õnnestus selline kapseldamine ilusti. Varem läbi võetud teema näited leiti eelmistest kohtadest ilma suurema otsimiseta mälu järgi üles. Varasemat näidet uue teema alusena kasutades kopeerisin selle julgesti ka uue tunni kataloogi ning otsimise peale kuigivõrd aega ei kulunud. Kordavate ülesannetega tunnis uut kataloogi ei loodud ning tuli eelmiste hulgast sobivad näited leida. Selline jaotus töötas ühtlase tasemega algajate rühma puhul. Edasi-jõudnutele sobisid temaatilised veebilehed. Suurima ja vähima ploki näidete arv erines rohkem kui kaks korda, kuid vähemalt oli enamik kohti nüüd loogilist jada pidi leitavad ja

ühele veebilehele seatud näidete arvu võis alamlehtedeks jagamise ning alapealkirjade abil kasutajale ülevaatlikus vahemikus hoida. Mõneteistkümne viite hulgast õnnestub peale vaadates enamasti sobiv üles leida. Kui aga lehel on näiteid üle kahekümne, peavad need olema alapealkirjade või muude eraldajatega kümnekonna näite suurusteks loogilisteks gruppideks eraldatud. Siis paraneb otsitud lõigu leitavus märgatavalt.

Näidete jagamine viie suure teema vahel ning otsingupuu kahetasemeliseks muutmine tegi struktuuri selgemaks ning kergendas sobiva koha leidmist nii otsimisel kui andmete ülespanekul, kuid jättis ikkagi lahtiseks mitme jaotisega seotud teemade leidmise. Paigutamisel seadsin teemad enese tarvis mõttes tähtsuse järjekorda ning kui näites oli kasvõi veidi puudutatud tähtsamat teemat, siis paigutasin näite sinna. Nii aga jäid jaotisest välja mitmed teemad, millega kaasnes lõike minu silmis tähtsamatest teemadest. Nii näiteks lõimede töö on siin peaaegu alati seotud võrguvahendite või graafikaga ning otsesesse lõimede peatükki jäänuks vaid paar terviklikku näidet, mis said seekord keele vahendite üheks alateemaks pandud. Kui kasutaja on juba näidete paigutusega tuttav ning harjunud, õnnestub tal tõenäoliselt mõne aja jooksul enesele sobiv näide üles leida või veenduda selle puudumises. Esimesi kordi näitesaidile sattudes aga võib peajaotistest välja jäänud teemal näite leidmine küllalt vaevaline olla või tuleb sellest aja kokkuhoiu huvides suisa loobuda.

4.7.1 Dubleerivad viidete lehed

Eelpool kirjeldatud olukordade lahendamiseks tuleks näidete paiknemise puu kõrvale luua lisalehed mitme ploki vahel jaotunud teemade tarvis. Nii on tehtud mitmetes otsingumootorites ning arvan, et selline teguviis peaks end ka näidete paigutamisel õigustama. Ühe lisalehena olen välja toonud lihtsad näited, et kasutaja ei peaks põhitõdede ja -vahendite otsingul sadades pikkades ja keerulistes seletustes tuhlama. Sarnaseid lehti peaks aga esialgsel hinnangul olema kümmekond, võibolla rohkemgi. Kui neid rohkem koguneb, on abilehedki ehk mõistlik kuidagimoodi gruppidesse koguda. Siiski arvan, et peaks kindlalt eristama andmete paiknemise põhipuud ning lisalehtede "abipuud". Kui mõnel inimesel on soovi põhipuud järjest uurides kõik näited üle vaadata, siis peab tal see võimalus olema. Viiteid omavahel segades võib samale näitele sattuda mitmest kohast ning ei anna selget ülevaadet, palju on läbitud, palju on ees.

4.7.2 Märksõnad

Otsilehtedes on suur osa automaatsel märksõna järgi otsingul. Hoolimata koodinäidete valdkonna spetsiifikast leian, et otsing sissetipitud sõna järgi annaks ka siin häid tulemusi, mõnel juhul võiks kiirendada näite leidmist või puudumisest teada saamist mitu korda. Kui näidetest koostada andmebaas ning iga näide varustada tema eri tahke kirjeldavate märksõnadega, siis õnnestub kergesti silma alla saada kogu vastava teemaga seotud näidete kogumik ning leida neist omale sobiv või veenduda selle puudumises.

4.7.3 Täistekstiotsing

Ka täistekstiotsing kuluks ära. Java loojad võivad küll kiidelda korraliku javadoc'i abil genereeritud pakettide, klasside, meetodite ja parameetrite kirjeldusega, mida igal keelel sellisel kujul olemas pole. Kui käskude juures olevatest seletustest väheks jääb, pole katsetamiskõlbulikke töötavaid näiteid kusagilt suurt leida. Muidugi võib uurida õpikuid ja lasta otsimootoritel vastavat käsku sisaldavad koodid silma ette rivistada, kuid sobiva õpiku ja sobiva näite leidmine võib võtta hulk aega ning pole kindel, kas vastava käsu kohta näidet koostatudki on. Interneti pealt leitud vastavat sõna sisaldavad failid aga ei pruugi üldsegi mitte töökõlbulikud programmid olla. Korrektse näitearhiivi piires aga peaksid kõik näited vähemasti tööle minema ning töötava käsu puhul on juba kergem seda koodi täiendada, parandada ning uurida, kuidas õnnestub leitud korraldus oma koodi sobivale kujule paigutada.

4.8 Kokkuvõte

Kokkuvõtteks leian, et programmeerimise õppimisel on töötavad näited tähtsaks abivahendiks. Sõltuvalt õppija tüübist võib näidete osa õppes olla lihtsast illustreerivast materjalist ja enesekontrollivahendist kuni kogu õppimise alustala ning põhilise toetuspinnani välja. Näidetele toetujale on väga tähtis, et töö aluseks võetud näited töötaksid korralikult, et neid pole lihtsustamise huvides käivitumiskõlbmatuks muudetud ning ilustamise tarvis nende põhisisu liialt ära peidetud. Sobiva näite leidmise lihtsus ja kiirus lubab keskenduda sisulisele õppimisele. Näidete leidmist aitavad lihtsustada loogiline sisuline hierarhiline jaotus, abilehed sulanduvate teemade leidmiseks, otsing märksõnade ning täisteksti järgi.

5 Õpimudelid ning -teed

Magistritöö autor on programmeerimist õpetanud 5 aastat (alates 1997), sellest Java ning Javaskriptiga seotud kursusi pidanud 3 aastat. Selle aja jooksul on nii teadlikult

andmeid kogutud kui spontaanselt tekkinud olukordi tähele pandud ja analüüsitud mitmekümne mõnest päevast kuni poole aastani kestva kursuse jooksul. Kõik järgnevalt pakutud võimalused on testitud ja järgnevalt analüüsitud vähemalt korra, mõnda on saanud aga rohkem kui kümme korda läbida ja viimistleda.

5.1 Java programmeerimiskeele algõpetus

Algõppe eesmärgiks võib olla nii huvi tekitamine programmeerimise vastu, üldise mõistmisvõime kasvatamine kui inimese ettevalmistamine tulevase programmeerimistöö tegemiseks. Esimene võimalus on väga tähtis isiksuse arenemise ajal ning tekkinud huvi ühe ala vastu võib edaspidi märgatavalt soodustada inimese võimet teemadesse süveneda. Sellest konkreetsete operatsioonide õpimudel. Kui eesmärgiks pole mitte lihtsalt programmeerimisega tutvuda ja valdkonna vastu huvi äratada, siis on aastate jooksul harjutud teemale süstemaatilisemalt lähenema. Nii loodetakse, et kursuse läbinud õppuritel tekib terviklik ettekujutus ning nad suudavad oma võimeid hinnata ning teavad, kust millisele probleemile lahendust otsima minna. Mitmete gruppide õpetamise kogemuse põhjal võin välja tuua järgevad kolm edukamat liini, mis paarikümnetunnilise õppimise järel annavad ligikaudu samamahulise tulemuse. Nimetaksin neid liine: käsureamatemaatika, käsureagraafika ning mitmekülgsed rakendid.

Liinid on algfaasides küllalt erinevad ning neid omavahel segades võib keskmised ja nõrgemad õppurid kergesti segadusse viia ning sellega tempos märgatavalt (kuni kaks korda) kaotada. Grupid pole üldjuhul Javaga varem kokku puutunud, kuid on programmeerimisega poole aasta jooksul tutvunud. Imestusega võib märkida, et varem programmeerimisega kokku puutunud või sellega mitte tegelnud gruppide puhul sobivad samasugused õpiteed. Esimesel juhul lihtsalt suudetakse samad etapid mõnikord kiiremini läbida ning mõnest kohast üle hüpata sama teemat sügavamalt käsitleva osa juurde. Üllatuslikult mõjutab lõpptulemust märgatavalt enam grupi üldine motivatsioon ja nupukustase kui eelnevalt läbitud programmeerimiskursus või õppijate vanus.

5.1.1 Konkreetsed operatsioonid

Programmeerimiseks pole mitte keegi liiga noor, tuleb lihtsalt valida õppimiseks ja õpetamiseks sobivad meetodid. Väljamaal on populaarsust kogunud programmeeritavad mänguautod, mille käsustik piirdub kümnekonna korraldusega ning kus põhitöö seisneb raja ettevalmistamises ning testides, kuid mille kaudu võib tekkida ka täiesti arvestatav

esmane programmeerimiskogemus. Ligilähedaselt samatüübilised oskused ja kogemused saab anda ka käskudega arvutit juhtides - programmeerimiskeele valik iseenesest siin liialt suurt tähtsust ei tohiks omada. Javas küllalt hõlpsasti kättesaadavad graafikavahendid täiesti võimaldavad põhikoolis programmeerimishuvi tekitada. Seda on autor nii koolipraktika kui programmeerimisringi raames proovinud. Huvi säilimiseks peab põhitähelepanu olema soovitud tulemuse (töötava rakenduse) saavutamisel, keele vahendid jäävad tugistruktuurina tahaplaanile ning nendega tuleb tutvuda vastavalt uuritava teema vajadustele.

Esmaseks lähtekohaks sobib mitmesugune paigutamine, liigutamine ja joonistamine. Vastavalt kujundite ja trajektooride keerukusele on põhjust uurida kooli matemaatika ning füüsika/loodusõpetuse raamatutest nii õpitud kui tundmatuid teemasid ning nende abil oma rakendusi täiustada. Nii nagu joonistustunnis saab teha ühiseid pilte või tehtut omavahel võrrelda, nii sobivad ka arvutil koostatud animatsioonid grupitööks või omavaheliseks võrdluseks. Tsükliteni jõutakse seetõttu, et mõne aja pärast tekib lastel idee oma töövaeva vähendamiseks, ning siis on nad vastavate näpunäidete suhtes hästi vastuvõtlikud. Samuti ollakse valmis meeldivate värvide koostamiseks uurima RGB värvimudelit või sama tegevuse varieeruvaks kordamiseks kasutama muutujaid. Mitme elemendi samaaegne liigutamine õpetab tegevusi koordineerima. Iseenesest on sellisteks ülesanneteks loodud LOGO nimeline programmeerimiskeel, kuid see ei tähenda, et muul juhul vastav õpetus ei võiks õnnestuda.

Teiseks kaasakiskuvaks tegevuseks on arvutiga dialoogi pidamine. Lapsed soovivad võimaluse tekkimisel päris keerulisi nii raamatutes/lehtedes ilmunud kui omamõeldud algoritme masinale arusaadavaks teha, et edaspidi valikute abil loodud maailmas ringi liikuda. Sellise süsteemi koostamise lihtsamaks teeks on hüpertekst, kuid programm võib tunduda vahel salapärasem. Eriti siis, kui lastakse näiteks juhuarvude abil olemasoleval süsteemil veidi muutuda. Seitsmendas klassis on küllalt palju põnevust pakkunud ka "Osta elevant ära"-stiilis sõnamäng, mida saab vastavalt oma fantaasiale edasi arendada.

Kogu sellise õpetuse lähtekohaks on: pigem vähem, aga paremini. Küll huvilised õpivad soodsa keskkonna abil ise edasi ning suudavad saadud teadmisi ka edaspidi rakendada. Huvi tähtsust rõhutab ka waldorfkoolitus [78]. Hea klappimise ning esialgsetest õnnestumistest tingitud huvi korral võivad põhi- ja keskkoolilapsed hakata ka teoreetilisemas osas ülikooli keskmisest tunduvalt kiiremini edasi liikuma. Nii kasvavadki

olümpiaadiülesannete lahendajad, täppisteaduste kooli õpilased ning loodetavasti ka tulevased edukad programmeerijad ning oma isikliku elu või muuteemalise töö tarbeks programme koostavad inimesed.

Siinsest materjalist sobivad sellise ligikaudu pooleaastase tutvumiskursuse tugimaterjaliks kolm-neli kuni mõniteist tutvustusnäidet. Oluliselt rohkem arvutialaseid õppematerjale taoline ring või kursus ei vajagi, sest põhienergia peab olema suunatud originaalsete ideede loomisele ja nende kaunile teostusele (või vahetevahel ka teostatavuse tõestamisele). Väga suur osa on õpetajal, kes peab oskama kiiret ja asjatundlikku nõu leida ja anda.

Konkreetsete tegevuste abil õppimine ei pea olema mitte ainult kainikute pärusmaa, ehkki psühholoogide määrangute järgi on selline suundumus vastavas vanuses tugevaim. Ka ülikoolis on mõnel programmeerimise vallas nõrgemal tudengil kergem ja arukam õppida selgeks konkreetsed toimingud ning nende põhjal edasises õppetöös tarvilikke ning mõnigikord mahukaid programmeerimistöid teha.

5.1.2 Käsureamatemaatika

Käsureamatemaatika liinis käsitletakse Javat alustuseks kui traditsioonilist objektideta programmeerimiskeelt. Pärast esmast teretava programmi käivitamist sobib suunduda näiteks ülesandeid genereeriva rakenduse koostamisele. Selle käigus tutvutakse tsüklite, valikute, muutujate, massiivide, juhuarvude ning alamprogrammidega. Andmesisestuseks sobivad käsurea parameetrid, kust on küllalt kerge sõnesid kätte saada. Nii tutvutakse ka tüübiteisendustega. Edasi tulevad failid ning sõnetöötlus. Üksikute teemadena sobib uurida keerulisemaid tingimuslauseid ning kahemõõtmelise massiiviga ümberkäimist teineteisesse sisenevate tsüklite abil. Samuti võiks koos massiividega käia java.util.Arrays'i sorteerimisvõimalus ning dokumentatsiooniga lähem tutvumine. Koos failidega peab paratamatult tähelepanu pöörama erinditele, lisaks aga sobib seletada muidki eriolukordi ning nendega toime tulemist. Seejärel võib rahulikult läbi võtta objektid, mängides läbi klasside loomise ja isendite kasutamise kirjetena. Meetodite lisamisega saab isendite "arukust" suurendada. Selguse säilitamisele aitab kaasa Java loojate soovitus hoida iga klass omaette failis. Siis võib kataloogi sisu vaadates kergesti näha, milliste klassidega tegemist on ning et iga klass on omaette sõltumatu üksus, mis võib teistele abi osutada. Vastavalt kursuse mahule, kasutatavale ajale ning grupi õppimisvõimele võib ette võtta nii pärimise, ülekatte, varje kui liidesed. Sisemised klassid

tuleks jätta hilisemaks, sest need kipuvad objektide selget pilti hägustama ning võivad takistada kindla süsteemi loomist, millele edaspidi toetuda. Näidete hulgas kasutatakse selle teema puhul traditsiooniliselt loomariiki ning sealset liikide hierarhiat.

Objektidega ühele poole saanud, jõutakse selleks ajaks enamasti ka keele süntaksi ja abiinfoga piisavalt harjuda ning edaspidi võib juba rahulikult ning vabalt valitud järjekorras vajalikke teemasid ette võtta kartmata, et mõni uus osa eelmised hullusti segi paiskaks. Sellise skeemi järgi on kirjutatud nii Jüri Kiho kui ka Ivor Hortoni õpikud (mõlemad viited). Kui siinseid materjale hakata lugema algusest, siis on püütud ka lähedase raja peal püsida.

5.1.3 Käsureagraafika

Siin asutakse nii keele ülesehitust kui käske tutvustama graafikavahendite kaudu. Päris algus võib olla küllalt sarnane konkreetsete operatsioonide liiniga, kuid õppijatelt eeldatakse kõrgemat üldistusvõimet ning analüüsioskust. Kui esmane tutvus sobib teha main-meetodis liigutatava loodud standardraamiga, siis vähemasti teisest-kolmandast tunnist tuleb asuda sobivat raami alamklassi kokku panema. Liigselt kaua staatiliste meetodite küljes püsimine võib tekitada halva harjumuse kõike objektiväliselt lahendada, mis aga käib vastu Java ideoloogiale ning tekitab hiljem raskusi vähemasti sündmuste ja kuularite juures.

Järgnevalt tasub algoritmide koostamise oskusi treenida jooniste koostamisega, lubades kasutajal mõned parameetrid ette anda. Lihtsamaks treeninguks on soovitud suurusega lihtsa kujundi koostamine. Edasi aga võib pakkuda näiteks soovitud korruste (ja trepikodade) arvuga korrusmaja koostamist (kahekordne tsükkel), samuti tulpdiagrammi loomist ning maailma- ja ekraanikoordinaatide ja mõõtkava arvestamist ning skaala joonistamist.

Erinevalt veebis või õpikutes levinud näidetele, kus oma programmid luuaks Frame alamklassina, olen paremaks pidanud Panel'it või Applet'it. Sel juhul on loodud klassidest isendeid kergem hiljem muude konteinerite sisse või veebilehele paigutada. Käskude arv sellest oluliselt ei kasva, sest mis muidu tehakse Frame konstruktoris, sellest tuleb osa lihtsalt main-meetodisse üle kanda.

Sõneteisenduste harjutamiseks piisab kahest tekstiväljast - üks sisestuse ning teine tulemuste jaoks. Mitmesuguste sõnevahendite katsetamise puhul on otstarbekas selguse huvides pea igaühe tarvis eraldi klass teha.

Siiani piirduti ActionListeneriga, nüüd õnnestub kuularitest laiem ülevaade anda ning lisaks lihtsale toimimise konstateerimisele võib seletada ka töötamispõhimõtet. Samuti on õppurid juba piisavalt vastuvõtlikud enamike graafikakomponentide tegevusala mõistmiseks ning suudavad ka paigutamisega ilma suuremate probleemidega hakkama saada. Koos hiirekuulariga annab koostada liikumist näitavaid rakendusi, samuti õnnestub küllaltki lihtsate vahenditega kokku panna graafikaredaktor, mis kasvatab õppurite enesekindlust ja usku õpitavasse keelesse. Kõrgema motivatsiooniga aga liigub kõik libedamalt. Mahukama kursuse puhul võivad liikumise juurde sobida ka lõimed ning isegi topeltpuhverdus. Ka omaloodud graafikakomponendi koostamine on küllaltki arendav teema, kuid selle juures peab hoolega silmas pidama, et õppurid aru saavad, mida nad teevad, muul juhul võib segadust enam olla kui tulu.

Käsureagraafika on algõppeks küllalt universaalne lähenemine ning ka siinsete materjalide puhul on tutvustusnäidete juures seda suunda silmas peetud. Kuna siinsel juhul ei pea suhtlema seiluriga, on üks lisaprogramm ning koos sellega tehniline probleem vähem. Samuti pole vaja luua lisaks HTML-faili. Ometi on enamik graafilise osaga alustanud õpikuid hakanud peale pigem rakenditest kui käsureaprogrammidest.

5.1.4 Mitmekülgsed rakendid

Rakendite puhul on sageli motivatsiooniks interaktiivsete veebilehtede loomine. Veebimänge ja muid rakendusi ollakse näinud ning nende koostamist tahetakse ka järele proovida. Mitmel kursusel on inimesi, kes on juba ette palunud et õpetataks veebilehe applettide tegemist. Flash ja muud vahendid on veebilehtede programmeerimisel arvestatavatena rakendite kõrvale tulnud, kuid rakendeid peetakse endiselt küllalt võimsateks ja tähtsateks.

Ehkki edasine rakendite kaudu õppimine on küllalt sarnane käsureagraafika omaga, vaadatakse neid kui täiesti erinevaid maailmu ning katsed kohe algul eri pooli ühendada on vähemasti nõrgemate õppijate puhul lõppenud segadusega. Rakendi puhul tundub rakendikäituris oleva intepretaatori olemasolu arusaadavam kui käsureaprogrammide puhul, kus ollakse harjunud arvama, et vaid vastava operatsioonisüsteemi jaoks kompileeritu (windowsis nt. .exe) on "õige programm". Eriti tugev oli selline algarvamine üheksakümnendatel, kui Java vähem tuntud oli. Rakendi puhul tuleb juurde HTML-i fail, mis lisatükina algset arusaamist ja tükkide

süstematiseerimist kipub raskendama. Mõne tunni jooksul veebilehtedega tutvust tehes ning siis lisades sinna ühe komponendina rakendi, pääseb komistuskivist mööda.

5.1.5 JSP põhine õpe

Lihtne alustada, sest korralikult ettevalmistatud keskkonna puhul pole vaja eraldi kompileerimist, HTML-i sisse saab käsud kirjutada lihtsasti. JSP sobib näiteks Javaskriptiga kokku puutunutele lihtsaks veebiserveri programmeerimise tutvustuseks ning ka Java keele näitamiseks. Lihtsast paaritunnisest tutvustusest ei tasuks algajatel kaugemale minna, sest saabuvad veateated tulevad vahepealse servlettideks kompileerimise tõttu küllalt arusaamatud ning häirivad. Lihtsamaks ja kiiremaks veebiserveri programmeerimise õppimiseks sobib PHP või ASP paremini. JSP juurde võib hiljem ringiga tagasi tulla, kui servlettide tööpõhimõtetest on arusaamine tekkinud ning samuti osatakse eraldi teovõimelisi abiklasse koostada, et neid siis JSP juures ubadena tööle panna.

5.1.6 Servlettide põhine õpe

Enamasti jõutakse servlettideni kursustes või õpikutes alles pärast pikemat muude teemadega tutvumist, samas ei pruugi algajale servleti loomine ja käivitamine sugugi keerulisem tunduda kui harilikust interpretaatorist väljundi vaatamine. Tulevaste andmebaasiprogrammeerijate koolitamisel alustati katsena ühes kursuses kohe servlettidest ning tulemused olid täiesti lootustandvad [58]. Märkimisväärne osa õppijaist ei olnud varem kuigi palju programmide kirjutamisega kokku puutunud. Kui on õppejõu abiga üle saadud esimestest tehnilistest raskustest (mis on paratamatud ka käsurealt töötava interpretaatori korral), siis brauseriaknaga on mõnigikord lihtsam andmeid vahetada. Jäävad ära DOS-i akna puhul tekkivad kooditabeliprobleemid ning HTML-vormist andmete pärimine pole samuti liialt keeruline. Nagu kogemused näitasid, võis küllalt julgesti edasi minna lihtsamate andmebaaside juurde ning saavutada häid tulemusi. Rõhutada tuleb sealjuures, et tegemist oli valitud seltskonna ning kõrge motivatsiooniga grupiga.

5.2 Edasijõudnute koolitus

Samas grupis olevate algajate puhul võetakse üldjuhul kõigiga läbi suhteliselt sarnased teemad. Ühed jõuavad teemadega sügavamale kui teised, kuid mõned aluskohad peavad omandama kõik, kes soovivad vastava ainega edasi minna. Mida kaugemale on jõutud üldpõhimõtetega, seda rohkem võib igaüks enesele spetsialiseerumist lubada ilma,

et lüngad mõne koha peal võiks edasisele õppimisele või tööle oluliseks takistuseks saada. Pigem on kasulik, kui igaüks enesele vajalikus või hästi õnnestuvas valdkonnas põhjalikumad teadmised saavutab, nii et võib nende abil omale võetud ülesandeid paremini lahendada. Spetsialiseerumise soodustamiseks olen leidnud järgnevalt kirjeldatud võimalused.

5.2.1 Grupiviisiline probleemilahendus

Õppijad jagunevad 2-4 liikmelistesse rühmadesse ning iga rühm valib/saab enesele ülesande. Neljalise rühma puhul võib tekkida segadusi, kus üks ei tea mis teine teeb. Viieliikmelises seltskonnas on juba väga raske hoolitseda, et keegi tööst välja ei langeks. Rühmas ülesannete jagamisel on võimalik arvestada osalejate eripäraga. Sageli leidub keegi nupukam, kes meeleldi võtab enesele raskemate algoritmide kokkupaneku, jättes suhteliselt tüütu kujundusliku töö teistele. Samas on mõnigi õnnelik, kui tal õnnestub mõne monotoonsema (kuid enamasti siiski ka mõtlemist nõudva) tööga teistele kasulik olla ning samas ka ise arenduskeskkonnas püsida. Igaüks grupist peaks kõigest koostatust aru saama ning vajadusel suutma kõikjale täiendusi teha. Tegemist on Extreme Programming'us olulise printsiibiga, mis võimaldab arendajatel oma kavandavat tööd vastavalt avastatud (tehnilistele) võimalustele ja takistustele kohandada.

5.2.2 Läbipõimitud materjal

Kui õppijaiks on kogenud programmeerijad, kes vajavad vaid vastava keele või teema suhtes täiendkoolitust, siis soovivad nad lühikese ajaga ammutada võimalikult suurel hulgal ning kõrgel tasemel enesel puuduvaid teadmisi ning seoseid. Pärast esimese tervitava programmi käivitamist tundub neile õigusega enamik tavaprogrammidest juba läbitud teena ning nende süstemaatiline läbiarutamine ajaraiskamisena. Sellisel puhul aitavad läbipõimitud teemadega näited ja ülesanded. Eks mingid teemad ristuvad omavahel alati, kuid edasijõudnud programmeerijate puhul pole vaja häbeneda näiteks sama põhja peal seletada ja õppida nii kasutajaliidese koostamist, andmebaasiühendust kui võrgukäsklusi. Kel mõni neist osadest pulkadeni tuttav, saab piiluda kõrvale veidi tundmatuma maa poole. Kui aga käsitletav teema juhtub ka kogenud programmeerijatele tundmatu olema, siis kehtivad samad reeglid mis algajatelegi, st. korraga tuleb tutvustada ainult üht tundmatut teemat ning see võimalikult lihtsalt välja tuua. Mõnedki õppevahendid kipuvad selle lihtsuse põhimõtte vastu eksima ning see tasub kusagilt

kurjalt kätte. Kui materjal jääb õppijale korraga haaratavaks, siis ei pea edasijõudnute puhul liialt lihtsustamisega tuumikut nudima.

5.2.3 Mitmetasemelised ülesanded

Koos õppivate inimeste arvutialased teadmised, oskused ja vilumused erinevad enamasti väga suurel määral. Ehkki hindamisskaalal üritatakse kõik positiivsed tulemused paigutada 50 ja 100% vahele, erineb tavalises klassis keskmiste õpilaste õppimisvõime paar-kolm korda ning äärmuste vahe võib kergesti olla vähemalt üheksa korda. Kes on jõudnud oma mõttetöös järgmisele abstraktsioonitasemele, selle produktiivsus võib hüppeliselt tõusta kolm korda, mõnedel juhtudel isegi terve suurusjärgu võrra. Minu hinnagul pole kooli ülesandeks mitte õpilasi arukuse järgi selekteerida, vaid igaühele kohast arenguvõimalust pakkuda, siis pean sobivaks jaotada lahendatavad ülesanded mitmesse tasemesse. Enamasti olen jaganud üheksakümneminutilise praktikumipaari kolme alamülesande lahendamiseks, kusjuures iga järgmine on eelmisest ligikaudu kolm korda mahukam. Ülesanded on püütud koostada nii, et kõige nutikamad ja töökamad jõuavad praktikumipaari jooksul napilt lahendada kogu kolmest osast koosneva suurema ülesande; enamik jõuab kolmanda osaülesande alguseni ning ka kõige nõrgemad saavad (väikese kõrvalise abiga) esimese osaga hakkama. Selliselt käsitlen õppijate hulka sisuliselt kui liitklassi, mis aga pakub igale võimaluse täie pingega kaasa töötada ning praktikumi jooksul maksimaalselt õppida. Sarnast süsteemi on kiitnud Johannes Käis juba 1935. aastal [35].

5.3 Veebiprogrammeerimise kursuste komplekte

Tahtes anda õppijatele mingi taseme ülesannete sooritamiseks terviklikku pilti, tuleb üksikute teemade ja keelte õpet omavahel kombineerida. Siiski tundub vähemalt esialgu, et on lihtsam õppida üksikute rohkem või vähem temaatiliste kursuste kaupa kui püüda kõike korraga selgeks saada. Allpool on välja toodud mõned järeleproovitud ning suhteliselt hästi õnnestunud komplektid.

5.3.1 Kliendirakenduste programmeerija

Käsureamatemaatika + HTML-kujundus + Javaskripti lahendused

Sellises järjestuses õpitakse peamiselt staatilisi meetodeid kasutavate programmide peal selgeks korrektsete programmide kirjutamine ning algoritmide koostamine. Java range süntaks sunnib arvestama muutujate tüüpidega ning ei lase naljalt läbi ülejala kokkupandud

programme. Esimese osaga ei taotleta särava tulemuse saavutamist, kuid võetakse läbi enamik teemasid, mis on vajalikud arvutustega seotud programmide koostamiseks. Lahendatakse hulga mõtlemist nõudvaid ülesandeid.

Järgnevalt asutakse veebilehti koostama ning tutvutakse veebikujunduse printsiipidega. Algul tasapisi, hiljem üha suuremal määral lisatakse lehtede interaktiivsemateks muutmisel Javaskripti koodi.

Kursuste komplekti läbimine ei nõua osalejalt liialt suurt taiplikkust, küll aga püsivust korrektse koodi loomisel ning veidi häid ideid veebilehtede kokkupanekul. Komplekti läbinu peaks suutma koostada asjalikke ning meeldivaid veebilehti ning lisada sinna nii näitlikustavaid kui arvutustöid teostada aitavaid programme. Samuti peaksid komplekti läbinud saama hakkama oma edasises koolitöös vajalike mitmeastmeliste arvutustööde automatiseerimisega. Sellise järgnevuse on läbi teinud TPÜ matemaatika eriala üliõpilased.

5.3.2 Laia profiiliga veebiprogrammerija

Tutvus Javaskriptiga (+ JSP) + käsureagraafika + veebitehnoloogiad

Mõeldud veidi rohkem arvutihuvilisemale õppegrupile ning tulemusena loodetakse laia silmaringiga veebiprogrammeerijaid, kes saavad hakkama nii graafiliste kui andmebaasipõhiste veebirakenduste koostamisega. Ka programmeerimise poolest nõrgemad sellest grupist peaksid suutma programme projekteerida ning ideede teostatavust analüüsida.

Võimalikud veebilehtede struktuurid arutatakse läbi kogemuse baasil. HTML-iga tutvutakse lähemalt Javaskripti kaudu. JSP-ga antakse semestri lõpul väike ülevaade veebiserveri programmeerimise ideoloogiast. Käsureagraafika abil tutvutakse põhjalikumalt graafilisi võimalusi nõudvate programmide koostamisega. Ehkki alustati graafikaga, võetakse ettetulevate ülesannete käigus läbi kõik Java põhikursuse teemad, kaasa arvatud failid, andmetöötlus ning võrgu pistikliides. Graafika pakub piisavalt alust ning näitlikkust andekate algoritmide koostamisega harjumiseks.

Veebitehnoloogiad võetakse ette vastavalt tehnilistele võimalustele, kuid andmebaasipõhiste süsteemide ehitamist vaadeldakse soovitavalt PHP ja/või servlettide näitel. Taas ringiga tagasi tulles uuritakse ka JSP pakutavat lahendust. Võrreldakse sarnaste programmide koostamist ASP ning PERLi abil. Silmaringi laiendamiseks võidakse tuttavaks õppida pea kõik Java jätkukursuse teemad - seda nii pilditöötluse kui

võrgurakenduste ja turvalisuse poole pealt. Sellise järgnevuse on läbi teinud TPÜ informaatikatudengid.

5.3.3 Andmebaaside veebiliideste programmeerija

Servletid + käsureamatemaatika + veebitehnoloogiad

Võrreldes eelmise jaotisega on välja jäetud pea kõik graafikaga seonduv. Keskendutakse andmehulkade ning andmebaasidega seotud programmeerimisvõtetele ning keerukamate algoritmide koostamisele. Seetõttu saavutatakse tööleasumiseks kõlbulik tase vähem kui aastaga ehk poole rutem kui eelmisel juhul. Servlettidega alustamine nõuab algul suuremat tähelepanu, kuid kuna nendega tuleb vastavatel õppuritel tulevikus enam tegemist, siis tasub pingutus end ära.

5.3.4 Veebilehtede koostaja

Veebiredaktorid + HTML-nipid + lihtsad skriptid + kohandamine + PHP

Tegemist lihtsama järjestusega, kus usin, arvutiga kokku puutunud, kuid programmeerimisvõõras inimene suudab üldjuhul saavutada igapäevatöös vajalike meeldiva väljanägemisega veebilehtede kokkupanekuks tarvilikud oskused ligikaudu neljakümne tunniga. Usinamad suudavad koostada ja kohandada lihtsamaid skripte ning aru saada veebiserveri programmeerimise võimalustest ja võimatustest. Nõrgemad piirduvad edaspidi staatiliste lehtede kokkupanekuga, kuid suudavad mõista Javaskripti võimalusi ning ligikaudu hinnata selleteemaliste tellitavate tööde mahtu.

5.4 Õppematerjalide kasutajad

5.4.1 TPÜ informaatikatudengid

Enamik kirjapandud teemasid leiab ühel või teisel viisil kasutust Tallinna Pedagoogikaülikooli informaatikakursustes. Eelkõige on konspektid ja näited koostatud üliõpilaste jaoks. Valikkursuste tarbeks tuleks mõnda teemat süvendatumaltki käsitleda, kuid sellega kaasneb oht muuta materjalid raskesti haaratavaks.

Tekste on loengute jälgimisel hea kõrvalt vaadata, tähtsamad kohad alla joonida ning vajadusel mõned seletused ning uued võimalused juurde kirjutada. Samuti peaks neist konspektidest olema võimalik keerulisemate näidete juures näpuga järge ajada ning segased kohad enesele selgeks teha. Ka tudengid, kes pole saanud loengust osa võtta, peaksid materjalide abil iseseisvalt töötades suutma aine tempoga kaasas püsida.

5.4.2 Informaatika kõrvalainena

Kasutatavad materjalid sõltuvad õppekavast ning valitavatest ainetest. Informaatikat kõrvalainena õppivatele tudengitele on Java põhikursuse ja Javaskripti teemad olulisemad kui jätkukursuse omad. Kuna enamasti on tegemist inimestega, kel programmi kirjutamine pole igapäevane tegevus, tuleks jälgida, et nende silme ette ei satuks liialt palju häirivaid kõrvalvõimalusi. Kõigepealt tuleb omandada põhi, millele toetuda, ning alles siis suunduda lisadele, kust usinamad võivad täiendavaid teemasid hankida. Selle rühma üliõpilasi aitavad eraldi välja toodud lihtsamad näited ning nende seletused.

5.4.3 Eesti üldhariduskoolid

Logifailide järgi paistab, et võrku riputatud materjalide vastu on huvi tundnud sageli mõneteistkümne, harvem aga mitmekümne üldhariduskooli kasutajad. Õpetajad teavad, et nii arvutitundides kui vabal ajal katsetavad huvilisemad koolipoisid mitmeid näiteid ning püüavad nende järgi omale meeldivaid rakendusi koostada. Tundub, et materjalide mõningase kohandamise ning koolide arvutihuvi kasvu puhul võib siinsete materjalide järgi õppijate ring märgatavalt laieneda. Õpetajate soovituste järgi tuleks lisada detailsed juhised lihtsamate programmide käivitamiseks ning "puust ette ja punaseks" selgitused ja KKK. Lisaseletused kuluksid ka muudele algajatele õppijatele, kuid varem programmeerimisega mitte kokku puutunutel ning eriti juhul, kui ka õpetaja end selles vallas tugevalt ei tunne, on sellised täpsed näpunäited lausa asendamatud. Samas on küllalt keeruline kirjutada algtutvustuse konspekti, kus osataks ette näha kõik olulisemad takistused ning jutt ei jääks tervikuna haaramiseks liialt pikaks. Siinjuures peab arvestama asjaolu, et koolipoisid asuvad seletustest abi otsima alles viimases hädas. Kui hädasolijal seal midagi raske leida ja arusaamatu lugeda on, kipub usk juhenditesse sootuks kaduma.

5.4.4 Arvutifirmad ja programmeerijad

Mõnelgi korral on siinsetest õppematerjalidest abi leidnud kutselised programmikirjutajad. Kui inimesed tegelevad mitme keelega, siis ei jõuta ühe keele peensusi enesele alati selgeks teha. Töö aluseks sobilik algnäide aga suudetakse hõlpsasti oma oludele kohandada. Niisuguste kasutajate jaoks kirjutades pole vaja keerukust karta. Konteksti tundes suudavad nad mõtte enamasti lennult haarata, vajadusel kiiresti muudest allikatest abiteavet hankides. Materjali paremaks leidmiseks kuluks näidetele lisada otsingusüsteem, kus õnnestuks käsku sisaldavad koodilõigud kiiresti silmade ette manada.

5.4.5 Eesti teised ülikoolid ning rakenduskõrgkoolid

Alates 1998ndast aastast on Java keele tähtsus programmeerimise õpetamisel kiiresti kasvanud. Tartu Ülikoolis, Tallinna Tehnikaülikoolis ja IT Kolledžis kasutatakse seda nii algõppe kui keerukamate kursuste raames. TPÜs asutakse Java juurde pärast vähemalt semestripikkust programmeerimisega tutvumist. Kõikjal on selle õpetuse tarbeks loodud oma materjale, samuti otsitud kokku välismaisi viiteid. Sellegipoolest on kõik seni eesti keeles võrku riputatud materjalid piisavalt erinevad, arenemisel nad vaid täiendavad teineteist. Magistritöö raames TPÜs koostatud materjalid on püütud teha lugemiseks ja mõistmiseks võimalikult selged ning tundub, et vähemasti lihtsamate teemade juures on see ka keskmisest paremini õnnestunud. Sellegipoolest on kohti, mis esmakordsel lugemisel arusaamatusi tekitavad ning mida oleks võimalik selgemalt kirjutada ning jooniste/näidetega varustada, nii et omandamine ladusamalt läheks. Lihtsa alustuse kirjutamine on ka maailmas levinud materjalide juures probleem. Algajatel tekib sõltuvalt tema kogemusest arvutiga ning kasutatavast riist- ja tarkvarast kümneid ja sadu probleeme, millele kompaktset kirjeldust ja lahendusi pakkuda pole hõlpus.

5.4.6 Abimaterjalid lühikursuslastele

Piisava hulga materjalide olemasolul saab koostatud konspektidest vastavalt vajadusele sobivaid alamtervikuid luua. Paarikümnetunnise täienduskursuse tuumikuks sobib mõnekümneleheküljeline tekst pluss kuni samapalju näiteid, mille abil teemaga tutvuda ning mille pealt tarvilikke rakendusi ehitada. Siin esitatud materjalide põhjal peaks väikeste täienduste abil saama lisaks kolmele suuremale jaotisele kokku panna järgmised lühikursused:

- Veebilehtede koostamine
- Javaskriptiga programmeerimine
- Java algkursus
- Java graafikavahendid
- Java muusikavahendid
- Servletid ja JSP
- Java ja XML

Materjalide koostajal enesel on küllalt lihtne sobivad lõigud kokku otsida ning nende põhjal kuulajatele soovitud kursus pidada. Parema struktureerimise korral suudaksid ka

muud õpetajad sealt hõlpsamini vajaliku alamosa leida ning selle abil materjale täiendada või tunde läbi viia.

5.4.7 Välismaised kasutajad

Kümnekonnal korral olen saanud välismaiste listide lugejatele mõne probleemi lahendamiseks pakkuda viite omakoostatud näitele. Logisid uurides selgub, et piiritaguseid lugejaid on rohkemgi - ka siis kui otsimootorid välja arvata. Eestikeelsed seletused on vaid võõrkeelt mõistvatele inimestele kasutud, kuid koodinäited koosnevad peamiselt Javakeelsetest lausetest ning on samuti kasutatavad kui näited muudestki allikatest. Sealjuures on näited enamasti piisavalt lühidad, et neist oleks võimalik aru saada ka ilma kommentaarideta.

Otsest kasu välismaised lugejad Eesti haridustasemele ei too, kuid leian, et siit näidete leidmine võib kergitada nende arvamust Eesti IT taseme kohta ning see omakorda suurendab siinsete ettevõtete/programmeerijate võimalusi välismaalt tellimusi saada.

5.4.8 Iseõppijad

Ametliku koolihariduse kõrval leidub alati inimesi, kes omal käel püüavad mõnda teemat enesele selgeks teha. Programmeerimisvaldkond on praegu küllalt populaarne ning see meelitab teemaga tutvuma inimesi teismelistest pensionärideni. Nende vajadused on sarnased üldhariduskoolide õpilastele, kuid mitmetel neist pole ka võimalik kooli arvutiõpetajalt nõu küsida. Seda tähtsamaks muutub iseõppimiseks tarvilik samm-sammult seletus, mille korralikuks koostamiseks peaks aga tõeline meister olema.

6 Kokkuvõte

Magistritöö käigus valmis tervikliku veebiprogrammeerimise kursuse metoodika ja töötati välja vastavateemalised õppematerjalid.

Metoodika puhul leiti, testiti ja lihviti sobivad õpiteed mitmesuguse ettevalmistustaseme ning eesmärkidega õppuritele.

Nii personaalse tagasiside kui metoodika piloteerimise käigus saadud tulemused annavad julgust ka edaspidi näiteid lühikeste ja lihtsatena hoida, hoolimata mõnikord kaasnevast kesisemast "kaubanduslikust väljanägemisest". Õpetamise jooksul tehtud katsete käigus on selgunud optimaalsed programmeerimise algõpetuse teed, millest kõrvalekaldumisel võib õppe tempo ning õpitavast arusaamine märgatavalt langeda.

Valminud õppematerjalid on leidnud kasutust nii ülikooli kursustes kui väljaspoolgi. Isikliku tagasiside tulemusena on õnnestunud neid kohandada vastavalt õppijate vajadustele. Eriti Java põhikursuse konspektist on saanud multifunktsionaalne õppevahend nii päris algajatele kui ka varem programmeerimisega tutvunutele. See võimaldab sobiva juhendamise puhul õppida mitmesuguste õpistiilidega inimestel. Ülesannete kogu pakub jõukohast lahendamist nii algajatele kui edasijõudnutele. Näidete kogust leiab abi enamike veebiprogrammeerimises ette tulevate tehniliste probleemide lahendamiseks.

Nüüdsest on eesti keeles veebiprogrammeerimise õppimiseks põhilised materjalid olemas ning materjali mahu ja kvaliteedi kasvatamine ei pruugi enam tähendada olulist hüpet emakeelses veebiprogrammeerimise alases hariduses. Loomulikult tuleb neid materjale pidevalt kaasajastada; ka keerukuse taset ja haaret on võimalik ning põhjust tõsta. Lähiajal on pigem tähtis olemasolevate materjalide võimalikult otstarbekohane ning intensiivne kasutamine ning mahuka ja kvaliteetse õppetöö tagamine.

Summary

The main goal of the master thesis "Holistic Approach to Web Programming" is to develop a holistic approach for learning/teaching web programming in Estonian language. This has been accomplished in four phases:

- To analyse existing web programming learning materials in Estonia,
- Composition of the web programming course,
- Elaboration of the principles for creating efficient code samples,
- Development of learning material (four textbooks) and code samples for middle level web programming.

The textbooks composed are:

- Basic Java programming (127 p)
- Advanced Java programming (108 p)
- Javascript programming (47 p)
- Collection of web programming exercises (68 pg.)

Programming code samples consist of 1000 units and cover most situations and algorithms used in middle level web programming. The code samples for mid-level problems are written in Java and/or in Javascript language. Samples covering basic level web-programming problems are also made in other languages like ASP, PHP, XML, VRML, PERL and HTML.

In thesis there are developed and tested different possible ways to start programming in Java for:

- Practice-based programming,
- Mathematics based programming,
- Graphics based programming,
- development web applets.

The methodology developed allow to compose special web-programming courses like for:

- development of web pages,
- development of client applications,
- development of web interfaces for a database,
- development of complicated web applications.

As a result of the thesis the following hypothesis were stated:

- Effective code samples must be short and ready to run
- In the beginning learn programming is recommended to concentrate to one subject

Kasutatud allikad

1. Baldwin, R. Java and JavaScript programming. http://www.phrantic.com/scoop/toc.htm 04.2002
2. BERTHOU. JavaSide.com http://www.javaside.com/
3. Carroll, C LearnASP. http://www.learnasp.com/classic/ 04.2002
4. DevEdge Online. Documentation - HTML http://developer.netscape.com/docs/manuals/htmlguid/index.htm 04.2002
5. Duran, L, Wimpsett, K, Adams, M. Java 2 Complete. SYBEX Inc. 1999
6. Eck, D. J. Introduction to Programming Using Java http://math.hws.edu/javanotes/
7. Eesti Telefon. NETI. NETI: / ARVUTID JA INTERNET / WWW abivahendid. http://www.neti.ee/cgi-bin/teema/ARVUTID_JA_INTERNET/WWW/
8. Eesti Telefon. NETI. NETI: / ARVUTID JA INTERNET / WWW abivahendid / Java ja JavaScript http://www.neti.ee/cgi-bin/teema/ARVUTID_JA_INTERNET/WWW/Java/
9. Fleming, Jennifer. Web Navigation: Designing the User Experience. 1998
10. Goodman, D. Dynamic HTML. 1998,
11. Goodrich, M. T., R.Tamassia. 1998. Data Structures and Algorithms in Java. John Wiley & Sons.
12. Grosso, P. XSL Concepts and Practical Use http://nwalsh.com/docs/tutorials/xsl/xsl/slides.html
13. Haamer,L XML. TPÜ, 2001
14. Habibullin, I. 2001. Samoutsitel Java. St. Peterburg.
15. Hall, M. core SERVLETS and JAVASERVER PAGES. http://archive.coreservlets.com/
16. Hall, M. Java Programming Resources http://www.apl.jhu.edu/~hall/java/
17. Hallik, K. Perl. http://madli.ut.ee/~kaidoh/perl/perl_opetus.html
18. Harold, E. R. Java Lecture Notes. Polytechnic University in Brooklyn http://www.ibiblio.org/javafaq/course/ 04.2002
19. Heinle, N. Designing with JavaScript. Web Review Studio Series. 1997
20. Hook, B. 1995. Building a 3D game engine in C++.
21. Horton, I. 2000. Beginning Java 2. JDK 1.3 Edition. Wrox Press Ltd.

22. IDG Communications Company. JavaWorld.com. http://www.javaworld.com/ 04.2002
23. INT Media Group. Active Server Pages 101. http://www.asp101.com/ 04.2001
24. INT Media Group. Learn How to Write Java Applets and Use Them On Your Web Site. http://www.webdeveloper.com/java/ 04.2001
25. Jaeger, A. HTML-juhend. http://madli.ut.ee/~ajaeger/fra31.html 04.2002
26. Java Advanced Imaging API Home Page. http://java.sun.com/products/java-media/jai/index.html
27. JavaFAQ.nu. Absolute Java FAQ. Free book, tip, code, applets, tips, freeware. http://www.javafaq.nu/
28. Jürgenson, R. 1993 Programmitehnika. Tallinn.
29. Kaasik, Ü. Kiho, J. Koit, M. 1990. Kuidas programmeerida. Tallinn. Valgus.
30. Kala, L. Mis on ASP. Tallinn, 2001 (referaat) http://postikana.ttu.ee/~leone/Referaat-ASP.doc
31. Kiho, J. 2001. Java programmeerimise aabits. Tartu: Tartu Ülikool. Arvutiteaduse Instituut.
32. Kiho, J. Java programmeerimise aabits. http://www.cs.ut.ee/~kiho/progr/Aabits/
33. Kiho, J. Väike Java Leksikon. Tartu Ülikool, arvutiteaduse instituut 2000
34. Kikkas, K. 1998. WWW ja HTML. TTÜ.
35. Käis, J. 1992. Isetegevus ja individuaalne tööviis. 2. trükk. Toimetanud Ferdinand Eisen. Koolibri. Tallinn.
36. Lepikult, T. Programmerimiskeel Java. http://www.math.ut.ee/~toomas_l/ProgrammeerimineJavas.html 04.2002
37. Leppikson, V. Programmeerimine C-keeles. Tln : EMI kirjastus, 1993
38. Linntam, A. Kuidas teha oma programme süsteemile Windows '95 (ja 'NT): Visual C++ algajale : kahekäsulisest programmist oma kalkulaatorini. Tallinn, 1997
39. Linntam. A. Interneti Kodulehekülje tegemine. Tallinn, 2001
40. Lyon, D, A. Image Processing in Java. Upper Saddle River : Prentice Hall PTR, 1999
41. Microsoft Developer Network. Active Server Pages Guide. http://msdn.microsoft.com/library/default.asp?URL=/library/psdk/iisref/aspguide.htm
42. Moncur, M. Teach Yourself Javascript in 24 Hours. 2001
43. Murulauk, M. Perl - Practical Extraction and Report Language http://www.tpu.ee/~marko666/perl

44. Netscape Communications Corporation. JavaScript Documentation. http://developer.netscape.com/docs/manuals/javascript.html 04.2002
45. Norden, D. VRML. TPÜ, 2001
46. O'Reilly & Associates, Inc. Online Catalog: Java in Nutshell. http://www.ora.com/catalog/javanut3/
47. O'Reilly & Associates, Inc. The Source of Perl. http://www.perl.com/
48. Peltomäki, J. Java-ohjelmoinnin peruskirja, Teknolit Oy 1999.
49. PHP.ee http://www.php.ee/ 04.2002
50. PHP: Hypertext Preprocessor. http://ee.php.net/ 04.2002
51. Pori Korkeakouluyksikkö, Tamperen Teknillinen Korkeakoulu. Loengud, 2000
52. Põldoja, H. Cascading Style Sheets. http://www.tpu.ee/~hans/css/ 04.2002
53. Põldoja, H. Veebisaitide kujundamine. http://www.tpu.ee/~hans/proseminar/ 04.2002
54. Pöial, J. Java leht http://www.cs.ut.ee/~jaanus/java/
55. Pöial, J. Võrguprogrammeerimise elemente. http://www.cs.ut.ee/~jaanus/java/loeng/loeng7.html
56. Robinson, M, Vorobijev, P. Swing. http://manning.spindoczine.com/sbe/ Swing. 04.2002
57. Roos, C. Java alused. http://www.tpu.ee/~charles/javaalus.htm
58. Roos, C. Java kursus. Tallinn, 2001
59. Roosalu, M. Marc V: WEB. http://home.delfi.ee/~marc/WEB/ 04.2002
60. Sibola, A. WWW-liides Tartu Ülikooli õppeaineregistrile. http://www.softronic.ee/~aulis/ 04.2002
61. Sinivee, I. HTML ja JavaScript. Tallinn, 2000.
62. Smith, D. 1991. Concepts of Object-Oriented programming.
63. Sun Microsystems Inc. Java 3D API Home Page. http://java.sun.com/products/java-media/3D/index.html 04.2002
64. Sun Microsystems Inc. Java Media Framework API. http://java.sun.com/products/java-media/jmf/index.html
65. Sun Microsystems Inc. The Java Tutorial. http://java.sun.com/docs/books/tutorial/ 03.2002
66. Sun Microsystems Inc. The Source of Java Technology. http://java.sun.com/ 04.2002
67. Sun Microsystems. JavaTM 2 Platform, Standard Edition, v 1.4.0 API Specification. http://java.sun.com/j2se/1.4/docs/api/overview-summary.html

68. Sun Microsystems. JavaTM 2 SDK, Standard Edition Documentation http://java.sun.com/j2se/1.4/docs/index.html
69. Sun Microsystems. JDC Tech Tips. http://developer.java.sun.com/developer/JDCTechTips/
70. Sun Microsystems. Словарь Java-терминов. http://www.sun.ru/win/java/start/gloss.html 04.2001
71. Tammet, T. Programmerimine I. Java ja SQL. http://www.ttu.ee/it/vorgutarkvara/wai4040/
72. The World WideWeb Consortsium. http://www.w3c.org/ 04.2002
73. Trees, K. HTML-õpetus. http://liriel.org/doc/87. 04.2002
74. Trees, K. Soovitusi algajale veebiprogrammeerijale. http://liriel.org/doc/188 04.2002
75. Tõugu, E. Java Fenomen. http://www.ttu.ee/aa/1998/2/java.html
76. W3C. XHTML Basic. http://www.w3.org/TR/xhtml-basic/
77. W3Schools Online Web Tutorials. http://www.w3schools.com/ 04.2002
78. Vabaduskasvatus. 1992. Rudolf Steineri pedagoogika. Pilte ja jutustusi rahvusvahelisest waldorfkoolitusest. Frans Carlgren, Arne Klingborg. Eesti õppekirjanduse keskus.
79. Vain.J. Operatsioonisüsteem Linux. TCP/IP-võrgu küsimusi. http://www.ise.ee/cdrom/cd2/linux/sisukord/9sisu.htm
80. Wikla, A. Ohjelmoinnin perusteet Java-kielellä, OtaDATA 1998
81. Wittenburg, T. 1995. Photo-based 3D Graphics in C++ . New York 1995
82. VRML Tutorial. http://xarch.tu-graz.ac.at/publ/tutorial/vrml/lighthouse/toc.html
83. Võrk. Võrk. http://vork.kolhoos.ee/