TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Anett Klaanberg

Murelahendajate koostamine Tartu Ülikooli kursuse „Objektorienteeritud programmeerimine" tarbeks

Bakalaureusetöö (9 EAP)

Juhendaja Marina Lepp, PhD

Tartu 2020

Murelahendajate koostamine Tartu Ülikooli kursuse „Objektorienteeritud programmeerimine" tarbeks

Lühikokkuvõte:

Tartu Ülikooli kursus „Objektorienteeritud programmeerimine" on muutumas üliõpilaste seas aina populaarsemaks valikuks. Suuremast kursusel osalejate arvust on tingitud olukord, kus osalejatel on üha erinevam taust ning varasem programmeerimiskogemus, mis muudab kursuse keerulisemaks nii õppurite kui ka läbiviijate jaoks. Ennist on rakendatud vihjete komplektide ehk murelahendajate süsteemi MOOC-ide (ingl *massive open online course*) raames põhieesmärgiga vähendada abiliinile esitatavate korduma kippuvate küsimuste arvu. Käesoleva bakalaureusetöö peamine fookus on aga toetada murelahendajate abil üliõpilaste õppeprotsessi. Selleks loodi viie praktikumi tarbeks murelahendajad ning analüüsiti üliõpilaste tagasiside põhjal nende tarvilikkust „Objektorienteeritud programmeerimine" kursusel. Tulemile tuginedes võib öelda, et üliõpilased said loodud murelahendajatest kursusel õppimisel abi ning pidasid neid kasulikeks õppematerjalideks.

Võtmesõnad: murelahendaja, objektorienteeritud programmeerimine, pööratud õpe
CERCS: P175 Informaatika, süsteemiteooria, S270 Pedagoogika ja didaktika

Troubleshooters for course „Object-oriented programming" in the University of Tartu

Abstract:

The increasing number of participants attending an object-oriented programming course every year is resulting in bringing together a variety of different backgrounds from the field. Previously, the troubleshooters have been used mainly in massive open online courses (MOOCs) with an objective to decrease the amount of frequently asked questions. However, this thesis focuses on troubleshooters supporting students with their homework thereby gaining broader knowledge in object-oriented programming. To meet the objective, troubleshooters were created for fifteen homework assignments and analyzed based on students' feedback to find out whether the troubleshooters form for this course works. The results of the study proved undergraduates getting help from troubleshooters and finding it useful for them.

Keywords: troubleshooter, troubleshooting, object-oriented programming, flipped classroom
CERCS: P175 Informatics, systems theory, S270 Pedagogy and didactics

Sisukord

1. Sissejuhatus

„Objektorienteeritud programmeerimine" on õppeaine, mis on kohustuslik kõikidele Tartu Ülikooli bakalaureuse õppeastme nii informaatika kui ka arvutitehnika ja matemaatika eriala esmakursuslastele. Veel on aine esindatud matemaatilise statistika eriala vabalt valitavas arvutiteaduse suunamoodulis ning on populaarne valik ka füüsika, keemia, materjaliteaduse ning paljude teiste erialade üliõpilaste seas. Õppeaine on üles ehitatud pööratud õppe vormile ning eeldusainena on üliõpilastel kohustuslik läbida vaid sissejuhatav kursus programmeerimiskeeles Python [1]. Kursusele registreerunute arv on kasvutrendis. 2018. aastal oli ennast ainele registreerinud 209, 2019. aastal 258 ning 2020. aastal lausa 289 üliõpilast.[1] Nii suure, eri taustaga ning pööratud õppe vormis kursuse õpetamine ning ka sellel õppimine võivad endaga kaasa tuua mitmesuguseid probleeme.

Kuna mingil osal üliõpilastest puudub enne ülikooli varasem programmeerimiskogemus, siis võib programmeerimise loogika omandamine osutuda võrdlemisi keeruliseks. Objektorienteeritud programmeerimine on arvutiteaduses lävimõiste, mis tähendab, et tegemist on kontseptsiooniga, ilma milleta võib edasijõudmine kindlas valdkonnas pidurduda. Lävimõiste võib endaga kaasa tuua aga olulised struktuursed erinevused varasemalt õpitust, seega võib keeruliseks osutuda varasemate teadmiste integreerimine uutega või lausa eelnevatest põhimõtetest loobumine [2, 3].

On tehtud hulgaliselt uurimusi selgitamaks välja, mis võivad tekitada programmeerimise õppimisel raskusi. Uuringute ajendiks on enamasti programmeerimiskursusel esinev kõrge välja- ning läbikukkumiste protsent. Näiteks Tan, Ting ja Ling uurimusest selgus, et õppurite jaoks valmistab kõige enam raskusi programmi väljatöötamine kindla probleemi lahendamiseks, aga ka funktsionaalsuste jagamine õigete koodiosade vahel [4]. Põhja Küprosel asuvas Near East Ülikoolis 2018. aastal läbi viidud analoogsete uurimisküsimustega teadustöös esinesid võrdlemisi sarnased tulemused. Näiteks valmistas üliõpilastele raskusi probleemülesande sisu lahtimõtestamine matemaatilisel kujul, programmi tarbeks algoritmi väljatöötamine ning tihtipeale oli üliõpilastel keeruline aru saada suuremahuliste programmide loogikast [5]. Vastavat tagasisidet on kogutud ka Tartu Ülikooli „Objektorienteeritud programmeerimine" kursuse raames. Näiteks 2018. aastal korraldatud tagasisideküsitluses aine korralduse ja õpikogemuse kohta toodi korduvalt välja, et esitatud ülesannete püstitused on tihti segased ja/või raske on näha, kuidas saavutada ülesandes küsitut ehk üliõpilased tunnevad

[1] Informatsioon pärineb aine vastutavalt õppejõult

puudust selgematest ülesandepüstitustest ning eksisteerib vajadus rohkemate suunitluste järele.[2] Tartu Ülikooli arvutiteaduse instituudi poolt korraldatud MOOC-ides (ingl *massive open online course*) on varasemalt kasutatud analoogsete probleemide lahendamiseks vihjete komplekte ehk murelahendajaid [6] ning kuna need on saanud osalejatelt positiivset tagasisidet, siis otsustati neid rakendada ka „Objektorienteeritud programmeerimine" kursusel.

Bakalaureusetöö eesmärk on luua aine „Objektorienteeritud programmeerimine" tarbeks murelahendajad ja analüüsida nende kasulikkust. Murelahendajate eesmärk on aidata abivajavaid õppureid kodutöö korrektse lahenduseni ning suuremas pildis objektorienteeritud programmeerimise loogika omandamiseni.

Õppeaines „Objektorienteeritud programmeerimine" hakkab alates 5. praktikumist kodutööde esitamiste arv märkimisväärselt vähenema.[3] Kursuse esimeses pooles keskendutakse esmatähtsatele baasteadmistele ning on oluline, et õppurid vastavad teadmised kindlasti omandaksid. Murelahendajates käsitletakse teemasid laiemalt ning teise nurga alt ehk see võib aidata üliõpilastel põhitõdesid paremini kinnistada. Murelahendajad võivad aidata parendada olukorda, kus õppuri jaoks tundub kodutöö liiga keeruline ning antakse alla ilma edasist abi küsimata. Lisaks võivad murelahendajad aidata vältida ka häbitunnet, mida mõned üliõpilased võivad abi küsimisel tunda. MOOC-is rakendatud samale süsteemile tuginedes võib öelda, et lisaks kursusel osalejate aitamisele võib murelahendajate positiivseks mõjuks kujuneda ka õppejõududele korduma kippuvate küsimuste esitamise mahu vähenemine [6].

Bakalaureusetöö teises peatükis antakse ülevaade objektorienteeritud programmeerimise taustast ning seda stiili järgivast programmeerimiskeelest Java. Töö kolmandas peatükis käsitletakse pööratud õppe vormi, selle eeliseid ja puuduseid ning rakendamist erinevate kõrgkoolide näitel. Seejärel vaadeldakse Tartu Ülikooli kursust „Objektorienteeritud programmeerimine", mis tugineb kahele eelnevalt käsitletud teemaplokile. Töö viiendas peatükis kirjeldatakse valminud murelahendajaid, nende loomise protsessi ning kasutatavat keskkonda. Viimasena analüüsitakse kursusel osalejate tagasiside põhjal murelahendajate kasulikkust ning tuuakse välja võimalikud edasiarendused objektorienteeritud programmeerimise õpetamise efektiivsemaks muutmiseks Tartu Ülikooli bakalaureuse esimese kursuse üliõpilastele.

[2] Informatsioon pärineb õppejõu tagasisideküsitlusest aine korralduse ja õpikogemuse kohta
[3] Informatsioon pärineb aine vastutavalt õppejõult

2. Objektorienteeritud programmeerimine

2.1 Paradigma olemus

Objektorienteeritud programmeerimine on üks võimalikest programmeerimise paradigmadest. Seda stiili võib pidada pigem uuema põlvkonna käsitluseks. Olgugi, et alus pandi sellele juba 1960ndatel aastatel, siis populaarsust hakkas see koguma alles 90ndate paiku. On programmeerimiskeeli nagu näiteks C++, mis toetavad lisaks objektorienteeritusele ka muid stiile, kuid on ka vaid ainult objektorienteeritusele suunatud keeli nagu näiteks Java [7]. Nagu ka nimi ütleb, tugineb sellises stiilis programmeerimine objektidele, mis on omavahel tugevas seoses. Objektid on klasside isendid, millel on erinevad omadused, olekud ning käitumismustrid. Klass, millesse isend kuulub, määrab küll ära, milline on objekti olek ning käitumine, kuid ühte klassi kuuluvate objektide väärtused võivad olla erinevad [3, 8]. Kasutatavat klassihierarhiat võib võrrelda näiteks taksonoomiaga ehk sarnaste isendite liigitamisega kindlate tunnuste alusel [9].

OOP-i eripära pärilus võimaldab aga hästi programmi kui tervikut struktureerida, mida peetakse selle stiili üheks peamiseks eeliseks eriti juhtudel, kui programmid on väga suuremahulised. Pärilus tähendab ülem- ja alamklasside struktuuri, kus alamklassid omandavad ülemklasside omadusi. Veel üheks eripäraks on liidesed, mis aitavad kirjeldada, milline funktsionaalsus peaks objektil olema, mille klass teatud liidest realiseerib [10]. Olgugi, et objektorienteeritud programmeerimise eripärad annavad suuri eeliseid, on need aga ka üheks peamiseks põhjuseks, miks seda paradigma esimese stiilina kõrgkoolides ei õpetata. Üldjuhul õpetatakse programmeerimise põhitõed selgeks esmalt struktuurselt lihtsamates keeltes nagu näiteks C või Python. Juba olemasolevale programmeerimistaustale tuginedes on üliõpilastel võrdlemisi lihtsam aru saada ka objektorienteerituse kontseptsioonist [11].

2.2 Programmeerimiskeel Java

Java on objektorienteeritusel põhinev programmeerimiskeel, mida kasutatakse ka Tartu Ülikooli „Objektorienteeritud programmeerimine" kursuse raames. Java esimene versioon avaldati Sun Microsystems'i poolt 1995ndal aastal, mil objektorienteeritud paradigma rakendamine programmeerimises üha enam populaarsust kogus. Selle välja töötamise ajendiks oli veendumus, et järgmiseks oluliseks sammuks tehnoloogiavaldkonnas saab olema digitaalseadmete ja arvutite kasutajate liidu teke [12]. Java õpetamise põhjuseid on mitmeid, kuid üheks olulisemaks võib pidada selle populaarsust. The TIOBE Programming Community

index'i, mis hindab programmeerimiskeelte populaarsust, alusel on Java hoidnud 30.03.2020 seisuga esikohta juba alates 2015. aasta keskpaigast ning ka varasemalt pidevalt esirindel olnud, kuid mõningatel perioodidel on olnud tarvis koht loovutada teisele väga populaarsele keelele C. On oluline märkida, et TIOBE indeks ei hinda programmeerimiskeele kvaliteeti, vaid seda, millises keeles on kõige enam koodiridasid kirjutatud. Hinnang koostatakse nii kvalifitseeritud arendajatelt, erinevatelt kursustelt kui ka otsingumootoritest nagu näiteks Google või Amazon kogutud informatsiooni põhjal [13]. Populaarsust toetab ka olukord tööturul, kus näiteks 30.03.2020 seisuga cv.ee 247-st infotehnoloogia valdkonna tööpakkumisest 45 ehk ligikaudu 18% sisaldas endas märksõna „Java" [14]. Lisaks populaarsusele on ka teisi aspekte, miks peetakse Java kasutamist heaks ning miks ka Tartu Ülikoolis just sellele keelele nii tugevalt rõhku pannakse. Java kasuks räägib selle lihtsus võrreldes näiteks teise populaarse objektorienteeritud programmeerimiskeelega C++ ning ka turvalisus. Lisaks asjaolu, et seda on lihtne üle võrgu korraga hallata ning see on arhitektuurist sõltumatu ehk seda saab käivitada iga Java virtuaalmasina abil [15].

3. Pööratud õpe

3.1 Õppevormi olemus

Pööratud õppe vormi (ingl *flipped classroom*) tutvustasid esmakordselt Baker ja Lage 2000. aastal. Laiemas mõttes tähendab pööratud õpe seda, et õppija omandab uued teadmised esmalt iseseisvalt ning alles seejärel toimub teema käsitlemine klassiruumis [16]. Kindlaid kriteeriumeid sellele aga määratud ei ole ning üldjuhul modifitseeritakse vormi vastavalt läbiviidavale kursusele. Pööratud õpet peetakse sobivamaks lahenduseks just väiksemate gruppide puhul [17]. Kui tegemist on aga suure kursusega, siis on üheks võimaluseks toetada praktikumides rakendatavat aktiivset õpet ka massloengutega, kus lähenetakse teemadele teise nurga alt ning selline meetodite integreeritus võib aidata kaasa teemade paremini omandamisele.

3.2 Eelised

Pööratud õppe üheks eeliseks on võimalus rakendada auditoorse tunni raames aktiivset õppemeetodit, sest teema läbitakse esmalt iseseisvalt kodus, kuid tavavormis õppe puhul tutvutakse teemaga alles praktikumi ajal. Aktiivne õpe aitab arendada ühest küljest rühmatöö oskust, teisalt kinnistada juba varasemalt õpitut läbi erilaadi meetodite, kus tuleb kogu eelnevalt omandatut analüüsida ning rakendada erinevates situatsioonides [18].

Mitmetest uuringutest on selgunud, et spetsiifilisema oskuse omandamisele, nagu seda on ka programmeerimine, aitab kaasa pööratud õppe rakendamine. Näiteks Cochin University of Science and Technology (CUSAT) infotehnoloogia ja arvutiteaduse ning tarkvaraarenduse harude eri astmete üliõpilastel on kohustuslik läbida andmestruktuuride kursus programmeerimiskeeles Java, kuid objektorienteeritud keeltest on neil varasem kokkupuude vaid C++'ga. Uuringu käigus loodi abistav 10-tunnine kursus Java õppimiseks, kus rühmad loodi vastavalt õpilaste enda hinnangutel ning varasematel tulemustel baseerudes. Nõrgema rühma puhul rakendati pööratud ning tugevama puhul klassikalist õppevormi. Kursuse toimumise ajal tehti testid nii 5-ndal kui ka 10-ndal nädalal ning tulemustest selgus, et nii tugevam kui ka nõrgem grupp olid saavutanud sama taseme ehk pööratud õppel on positiivne mõju sedalaadi õpingutes. Uuringus tõdeti, et valim oli küll võrdlemisi väike, 42 üliõpilast, kuid lisaks saadud tulemustele toetavad pööratud õppe kasulikkust ka mitmed teised uuringud [18].

Ühe edasiarendusena on katsetatud ühendada pööratud õppe vorm probleemõppega (ingl *problem-based learning*). Iirimaa Riiklikus Ülikoolis (National College of Ireland) korraldatud katses, kus rakendati probleem- ning pööratud õppe integratsiooni tarkvaraarenduse moodulis, selgus, et säärane edasiarendus on oluliselt tõhusam. Nimelt kukkus testülesande läbi ehk tulemus jäi alla 40% integreeritud õppe rakendamisel vaid 1,9% üliõpilastest, pööratud õppe puhul oli läbikukkumiste protsent 24,5 ning traditsioonilise õppe puhul 28,3%. Analoogselt Cochin'i ülikoolis tehtud uuringuga peetakse rakendatud õppemeetodi positiivseks väljundiks just nõrgemate õppurite tulemuste paranemist. Iirimaa Riikliku Ülikooli uuringu põhjal aga märgatavat vahet puhtalt pööratud õppe ning traditsioonilise vormi vahel ei ole, kuid pööratud õppe grupp oli võimeline paremini valima õigeid meetodeid probleemi lahendamiseks, edukam järelduste interpreteerimises ning saadud tulemuste ettekandmises [19].

3.3 Puudused

Pööratud õppe negatiivseks küljeks võib pidada aga seda, et osa õppuritest ei pruugi koduse materjaliga varem tutvuda ning see võib pidurdada praktikumi õppetööd. 2015. aastal Qatari Ülikoolis (QU) korraldatud pilootprogrammis, kus rakendati pööratud õppe vormi programmeerimisaines, selgus lisaks üldisele positiivsele mõjule aga, et ligikaudu 20% üliõpilastest ei töötanud enne auditoorset tööd nõutud materjali läbi, mis omakorda mõjutas nende saavutusi aktiivsel õppel ning seetõttu segas ka üldist õppetööd klassiruumis [20].

3.4 Üliõpilaste seisukoht õppevormi suhtes

Pööratud õppe positiivseteks külgedeks peavad üliõpilased seda, et teemasid käsitletakse rohkem süvitsi ning mitmel erineval viisil, mis omakorda aitab saavutada paremaid tulemusi ning õpiväljundeid edukamalt omandada. Lisaks tagab eelnevalt materjali läbitöötamine enesekindlama tunde auditoorseks tööks ning õpilastel on olnud praktikumideks valmistumisel võrdväärsed võimalused. Pööratud õppe puhul rakendatava aktiivse õppe kasulikkuseks peetakse veel seda, et õpitakse paremini probleemi lahti mõtestama ning infot teistele edasi andma. Negatiivse aspektina on välja toodud aga see, et iseseisvat tööd on rohkem ehk õpingutele tuleb panustada rohkem aega, kuid enamasti siiski tajutakse ka lisamahu kasulikkust [18, 19].

4. Tartu Ülikooli kursus „Objektorienteeritud programmeerimine”

4.1 Kursuse olemus

„Objektorienteeritud programmeerimine” on esimene nimetatud programmeerimise paradigma järgiv õppeaine, mis on kohustuslik kõikidele Tartu Ülikooli bakalaureuse õppeastme nii informaatika kui ka arvutitehnika ja matemaatika eriala esimese kursuse üliõpilastele. Sellel kursusel baseeruvad ka paljud edaspidised Tartu Ülikooli informaatika õppekava ained nagu näiteks „Algoritmid ja andmestruktuurid” ning „Tarkvaratehnika”. „Objektorienteeritud programmeerimine” õppeaine eeldusainena on üliõpilastel kohustuslik läbida vaid sissejuhatav kursus programmeerimiskeeles Python, kus lähenetakse programmeerimisele aga teise paradigma alusel [1].

Tartu Ülikooli „Objektorienteeritud programmeerimine” kursuse ülesehitus on võrdlemisi erinev klassikalisest mudelist. Õppeprotsess on jagatud kolmeks osaks: loengud, iseseisev töö ning praktikumid. Iga teemaploki kohta toimub esimesena loeng, mida on võimalik läbida kolmel erineval viisil: auditoorne loeng, kus kasutatakse klikkereid osalejate töösse kaasamiseks, virtuaalne loeng, milles osalemise tõendamiseks tuleb vastata iganädalastele küsimustele Moodles või edasijõudnute grupp, kus primaarsed teemad võetakse kiiremini läbi ning teemade hulk on laiendatud. Viimase grupi valivad enamasti üliõpilased, kellel on varasem programmeerimiskogemus. Pärast loengut tuleb õppuritel iseseisvalt läbi töötada praktikumimaterjal ja selle alusel lahendada enne kontakttunni toimumist kodutööd ning viimase sammuna toimub vastava teema praktikum.

4.2 Pööratud õpe rakendamine kursusel

Tartu Ülikooli „Objektorienteeritud programmeerimine” õppeaine on üles ehitatud pööratud õppe vormil, kuid suure kuulajaskonna tõttu on säilitatud ka massloengute toimumine [1]. Rakendatav metoodika sarnaneb oluliselt pööratud- ja probleemõppe ühendatud edasiarendusega. Näiteks on enamus paaristöid ning ka suurem projekt suunatud just reaaleluliste probleemide lahendamisele. Üldjuhul käsitletakse praktikumis põgusalt eelmise loengu teemasid ning seejärel lahendatakse praktikumiülesandeid aktiivse õppe vormis, millest kõige enam pannakse rõhku just paaristöödele (ingl *collaborative learning*) ning üksteise tööde tagasisidestamisele (ingl *peer review*). Veel toimub praktikumides kodutööde analüüsimine ning on võimalik õppejõult personaalsemat abi saada. Nagu ka teiste kõrgkoolide näitel võib ka Tartu Ülikooli kursuse puhul probleemiks kujuneda see, et üliõpilased ei tee endale selgeks

enne kontakttundi vastavate praktikumimaterjalide sisu ning ei osale auditoorses loengus või ei omanda vastavat materjali videoloengu teel. Kokkuvõtvalt aga selgus 2015. aastal Tartu Ülikoolis „Objektorienteeritud programmerimine" kursuse raames tehtud tagasisideküsitlusest, et kaks kolmandikku üliõpilastest on arvamusel, et pööratud õpe aitab õpingutele paremini kaasa kui seda teeb traditsiooniline vorm [16].

4.3 Kursuse õpiväljundid ja eesmärk

Kursuse „Objektorienteeritud programmeerimine" eesmärk on anda osalejatele baasteadmised vastava stiili spetsiifikast. Kursuse raames käsitletakse nii stiili eripärasid nagu näiteks kapseldamine, abstraktsioon, pärilus, polümorfism kui ka erinevaid rakendusteeke ning põgusalt ka andmestruktuure, millega tegeletakse aga süvitsi teisel õppeaastal kursuse „Algoritmid ja andmestruktuurid" raames. Lisaks õpitakse koostama suuremaid programme ning kursuse käigus tuleb luua paari peale oma projekt, mille eesmärk on arendada lisaks õpitu kinnistamisele ka rühmatööoskusi, mis tulevad tulevikus IT-alal töötades kasuks. Kogu kursusel omandatu loob tugeva vundamendi edaspidisteks õpinguteks, kuna suurem osa erialaseid kursuseid baseerub just objektorienteerituse paradigmal [1].

5. Murelahendajad

5.1 Abimaterjalide olemus

Troubleshooting on üks levinumaid probleemi lahendamise viise, kus esmalt tuvastatakse probleem ning seejärel antakse suunitlusi selle parandamiseks või tehakse vajadusel ise korda. Mõiste on kasutusel väga erinevates valdkondades, peamiselt küll infotehnoloogias, kuid ka näiteks inseneerias ja mehhatroonikas ning isegi psühholoogias [21, 22].

Murelahendaja (ingl *troubleshooter*) on varasemalt kirjeldatul baseeruv mõiste, kuid kitsamas mõttes käsitletakse seda käesoleva töö raames just programmeerimise õpetamise abivahendina. Murelahendaja on abisüsteem, mis aitab probleemile leida vastuse puustruktuurses süsteemis, kus igale probleemile on eraldi võimalik suunitlus saada, vajutades vastava vihje saamise nupule. Tartu Ülikooli arvutiteaduse instituut võttis need esmakordselt kasutusele 2016. aastal MOOC-is (ingl *massive open online course*), kus sellega kaasnes kohene positiivne mõju. Näiteks on uuringutest selgunud, et keskmiselt jääb MOOC-ide eduka lõpetamise protsent 15 ligi, kuid Tartu Ülikooli poolt korraldatud kursuse „Programmeerimisest maalähedaselt" läbivad edukalt iga-aastaselt üle poole osalejatest. Lisaks sellele vähenes abiliinile (ingl *helpdesk*) ehk meilile, millega lubati vastust vähemalt 8 tunni jooksul, saabunud kirjade arv lausa 29% võrra. 2016. aasta kevadel Tartu Ülikooli MOOC-is „Programmeerimisest maalähedaselt" osalejate seas läbi viidud vabatahtlikust tagasisideküsitlusest selgus, et ~80% osalejatest kasutas murelahendajaid ja 40,8% neist pidas murelahendajaid väga kasulikuks ning vaid 3,5% kasutajatest ei saanud nendest üldse abi [6]. Sellest võib järeldada, et suuremat kasutegurit omab see just kursuse läbijate seas, kes vajavad ülesannete lahendamisel rohkem suuniseid.

Murelahendajate negatiivseks küljeks võib aga pidada nende koostamise keerukust. Nii probleemsete kohtade tuvastamine, vihjete välja töötamine kui ka süstemaatiline organiseerimine võivad olla aeganõudvad ning komplitseeritud. Teisest küljest kui need on juba loodud ning ka tagasiside ja tulemuste põhjal edasi arendatud, siis muudab see kogu kursuse korraldamise palju lihtsamaks ning automatiseeritumaks. Veel võib murelahendajate rakendamine viia õpitud abituseni, kus murelahendajate poole pöördutakse juba ilma ise ülesannet lahendamatagi. Seetõttu ongi oluline, et murelahendajad oleks ühest küljest abistavad, aga teisalt ei annaks liiga palju abivajajale vastuseid ette. Üldises pildis kaalub aga kasutegur oluliselt üle puudujäägid [6].

5.2 Koostamise protsess

Murelahendajate loomise eesmärk kursuse „Objektorienteeritud programmeerimine" raames on aidata üliõpilastel kodutöö ülesande püstitust paremini mõista ning suunata abivajajaid kodutöö korrektse lahenduseni. Suurem eesmärk on aga aidata üliõpilastel omandada objektorienteeritud programmeerimise loogika, mis on baasiks nii edaspidistele õpingutele kui ka tugevaks konkurentsieeliseks tööturul. Varasemale tagasisidele tuginedes võib öelda, et õppurid tunnevad suunavatest abimaterjalidest puudust.[4] Käesoleva lõputöö raames koostati abimaterjalid 5 praktikumi tarbeks. Igas praktikumis on kolm kohustuslikku koduülesannet ehk kokku valmis 15 murelahendajat (Lisa I).

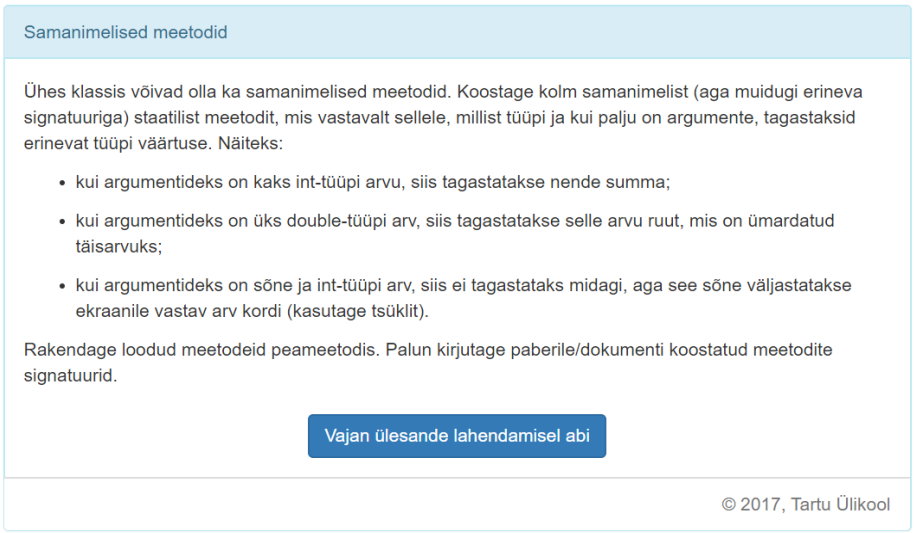
Murelahendajate loomise protsess algab sellega, et tuleb läbi töötada vastava praktikumi materjal, millele murelahendajad hakatakse koostama. See annab kõige parema ülevaate sellest, milline informatsioon on kättesaadav üliõpilastele. Läbitöötatud materjali põhjal saab kaardistada, mis võivad kursusel osalejate jaoks raskusteks kujuneda ning milline osa vajab rohkem lahtiseletamist. Murelahendajate koostaja jaoks võib katsumuseks kujuneda aga see, et ollakse ise kompetentne vastavas stiilis ning ei suudeta enam tajuda, millised aspektid võivad arusaamatuks jääda õppurile, kellel varasem kogemus puudub. Pärast materjali põhjal probleemide kaardistamist tuleb koostada vastavad suunavad küsimused ning vastused. Siinkohal võivad tekkida aga vastastikused situatsioonid, kus ühest küljest antakse abimaterjalide kasutajatele ette liiga palju informatsiooni, et õppur ei pea enam ise pingutama või teisalt on loodud murelahendaja, mille abi jääb vajaka. Kindlasti ühest õiget taktikat polegi võimalik välja töötada, sest iga murelahendaja kasutaja taust ja õppevõime on erinev ning selle põhjal tajutakse ka murelahendajate kasulikkust erinevalt. Murelahendajad tasub esmalt valmis kirjutada mõnes tekstitöötlusprogrammis, kus murelahendajate plokki on tervikuna võimalik parem hoomata kui seda on murelahendajate veebirakenduses. Kui murelahendajate koostaja on jõudnud järeldusele, et vihjete komplektid on valmis, siis tuleb need ka vastavasse keskkonda üles laadida. „Objektorienteeritud programmeerimine" kursuse raames võeti kasutusele https://progtugi.cs.ut.ee/ lahendus, mille töötas välja Vello Vaherpuu oma lõputöö raames MOOC-i kursuse murelahendajate tarbeks [23]. Tegemist on võrdlemisi universaalse platvormiga, mistõttu saab seda rakendada ka teistel kursustel.

[4] Informatsioon pärineb õppejõu tagasisideküsitlusest aine korralduse ja õpikogemuse kohta

5.3 Keskkond

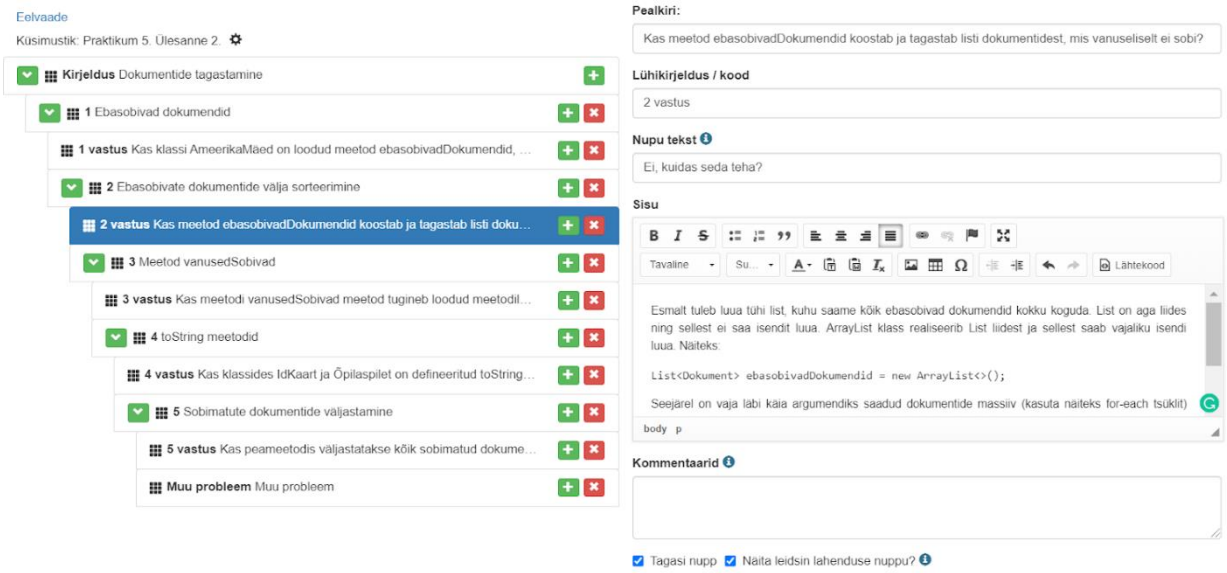
Nagu eelnevalt mainitud, kasutati vihjekomplektide jagamiseks õppuritega murelahendajate keskkonda, mis on leitav leheküljelt https://progtugi.cs.ut.ee/. Keskkonnal on kaks erinevat vaadet – kasutaja (joonis 1), kes saab hallata ning muuta üleslaetud murelahendajaid ning külaline (joonis 2), kes saab veebirakendust kasutada vaid abi saamiseks ning programmi töös muudatusi teha ei saa [23].

Joonis 1. Kasutaja vaade murelahendajate keskkonnas



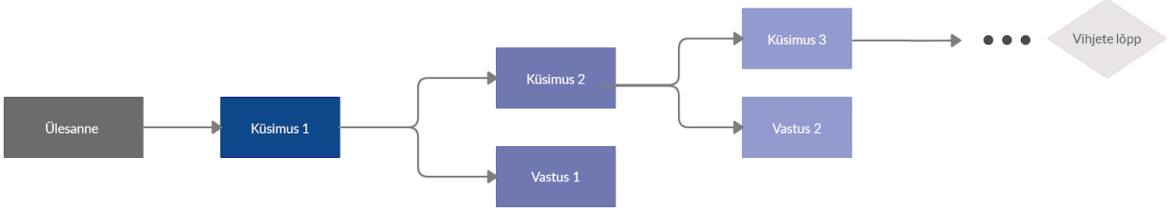
Joonis 2. Külalise vaade murelahendajate keskkonnas

Kasutajana saab keskkonda uusi murelahendajaid luua ning olemasolevaid hallata ja muuta, aga ka kloonida ning teiste samaväärsete kasutajatega murelahendajaid ning nende muutmise õigusi jagada. Joonisel 3 on näidatud uue murelahendaja loomise vaheleht. Iga samm tuleb puukujulises struktuuris eraldi sisse kanda. Sealjuures tuleb ära märkida küsimuse pealkiri ning sisu ja saab lisada ka lühikirjelduse ning kommentaare, et murelahendajaid oleks hiljem lihtsam hallata. Veel saab määrata nuppude valikud, et külaline saaks näiteks teada anda, et on vihjeplokist abi saanud või soovib tagasi minna eelmise sammu juurde.

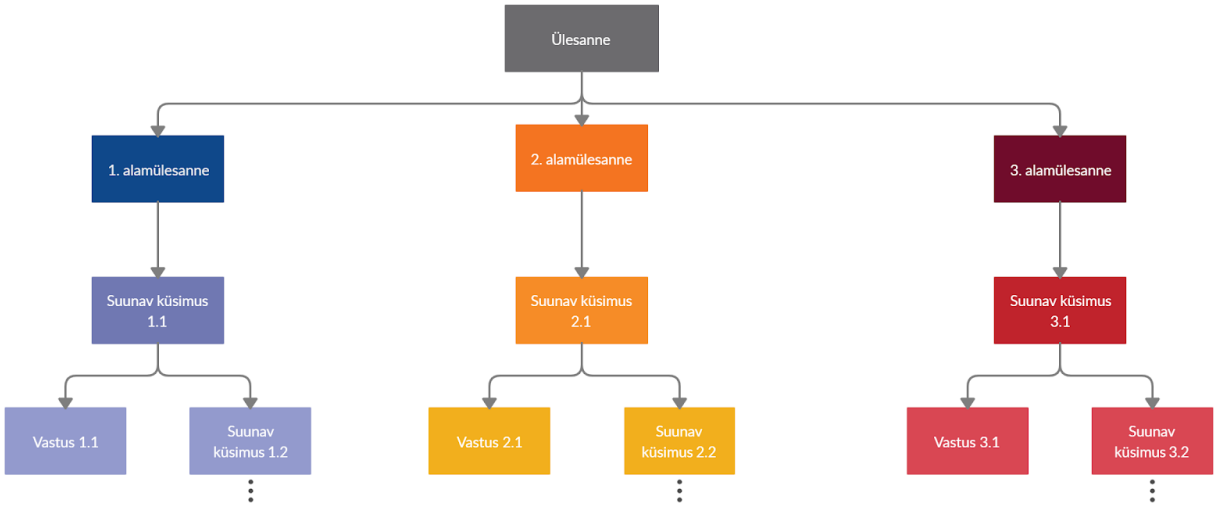


Joonis 3. Murelahendajate loomise vaheleht

Murelahendajaid on võimalik üles ehitada mitut moodi. Üldjuhul kasutatakse versiooni, kus igale küsimusele on määratud vastus ja sellele järgneb uus küsimus kuni jõutakse vihjete plokiga lõpuni (joonis 4). Kursuse materjalides esineb aga ka ülesandeid, mille korral on tarvis rakendada keerulisemat struktuuri (joonis 5). Näiteks kui ülesanne on jaotatud mitmeks alamülesandeks, siis võib esineda olukordi, kus õppur vajab abi näiteks vaid ühe kindla alamosa lahendamisel. Seetõttu on mõttekas murelahendajale luua vastavalt nii palju harusid, kui on alamülesandeid. Keerulisema ülesehitusega murelahendaja tasus luua näiteks 2. praktikumi 4. ülesande jaoks, kus tuleb koostada kolm erinevat samanimelist meetodit.

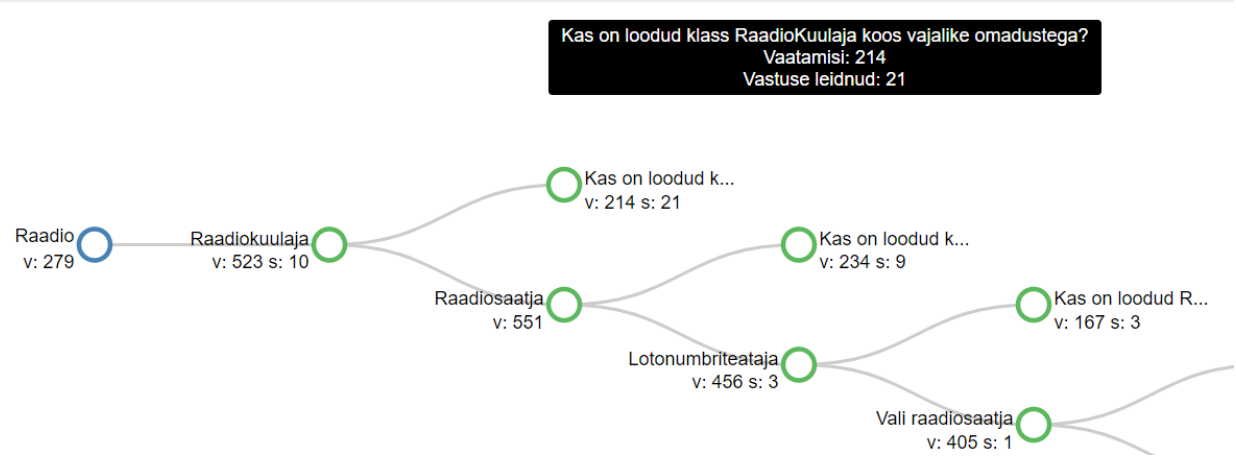


Joonis 4. Klassikaline murelahendaja struktuur



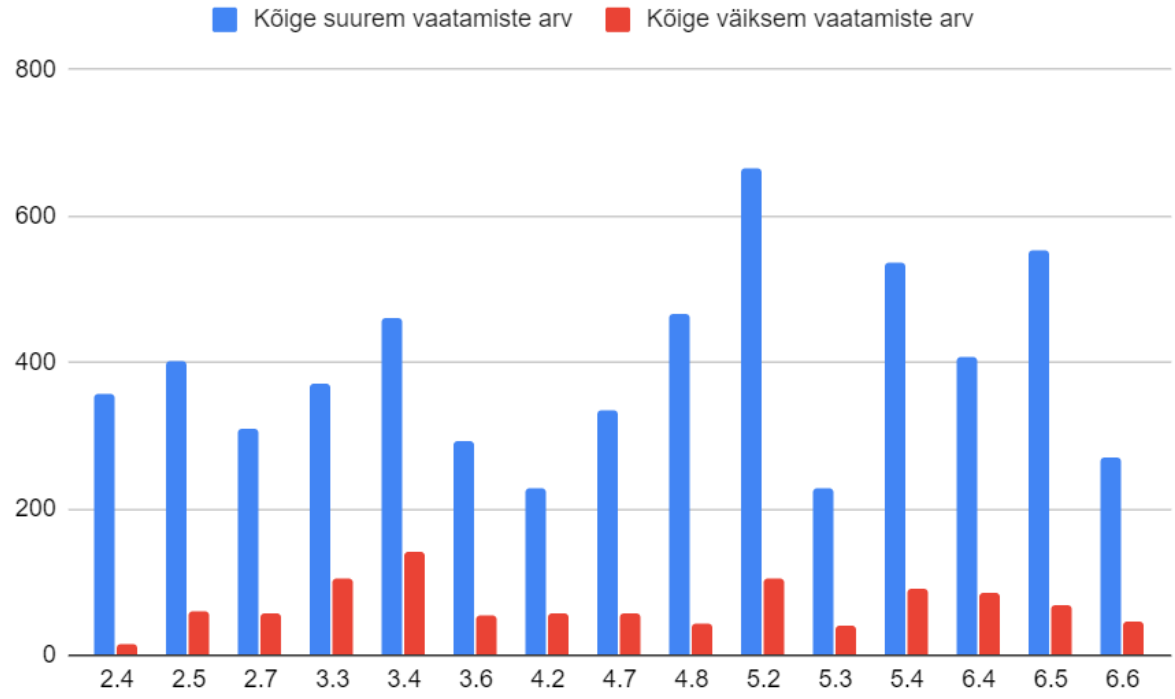
Joonis 5. Mitmeharulise murelahendaja struktuur

Lisaks sellele on võimalik kasutajana saada murelahendajate kasutamise kohta statistikat. Jooniselt 6 on näha, kuidas infot kuvatakse. Iga sammu kohta on võimalik teada saada, kui palju külalisi selle sammuni jõudis (joonisel tähistatud tähega „v") ning kui paljud leidsid kindla vihje juures vastuse (joonisel tähistatud tähega „s"). „Vastuse leidnud" kategooriasse läheb tulemus juhul, kui külaline vajutab vihje all olevat nuppu, et ta sai vastavast punktist abi. Siinkohal ei pruugi kuvatav informatsioon olla täielik, sest üheks variandiks pärast abi saamist on ka vaheleht sulgeda, mille korral vastuse leidnud külaliste arv ei suurene. Kui vihjetekomplekti lõpuni läbimisel ikka abi ei saadud, siis soovitatakse murelahendaja kasutajal ühendust võtta oma praktikumijuhendajaga.

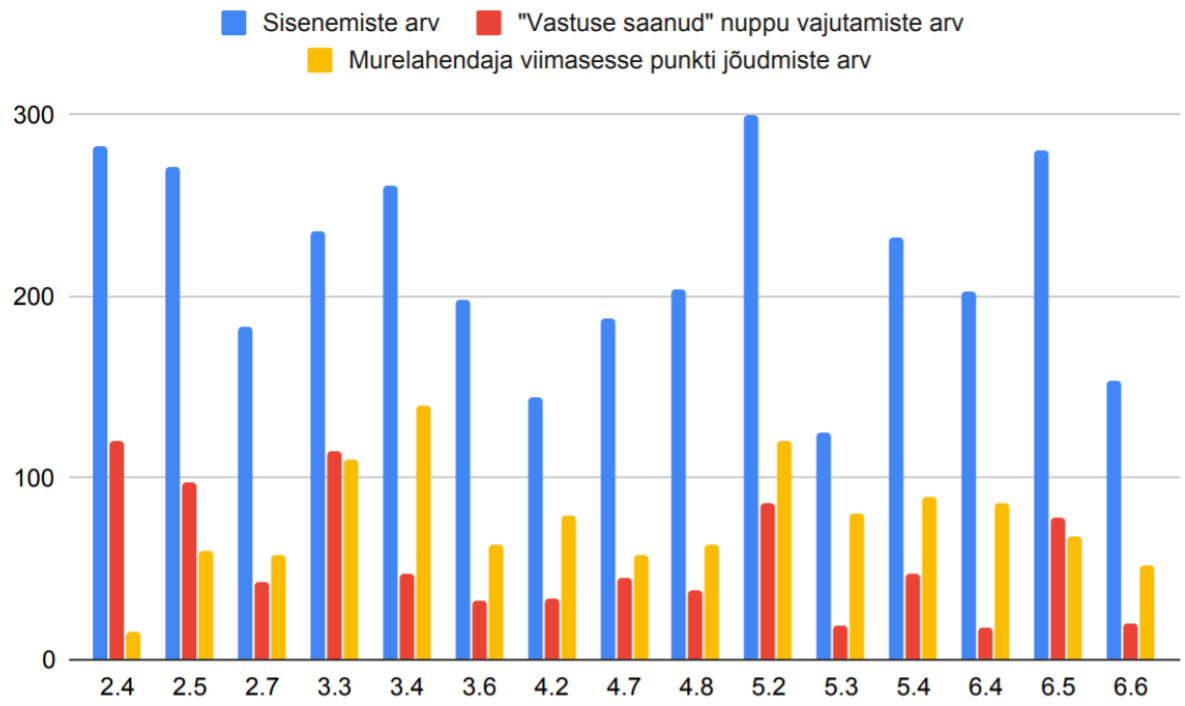


Joonis 6. Murelahendaja kasutamise statistika

Järgnevatelt joonistelt on näha käesoleva töö raames koostatud murelahendajate statistikat. Kõikidel joonistel on kasutatud ühesugust legendi, kus praktikumide ning ülesannete numbrid on esitatud vastava skeemi alusel („praktikumi number” „.” „ülesande number”). Joonisel 7 on kajastatud, milline oli iga murelahendaja kõikide sammude seast suurim ning väikseim külastamiste arv. Kõige enam vaadati 5. praktikumi 2. ülesande ühte vahelehte lausa 666 korda ning väikseim vaatamiste arv oli 2. praktikumi 4. ülesande ühe sammu korral kõigest 15 vaatamist. Jooniseid 7 ja 8 võrreldes on näha, et sisenemiste arv ei ole kunagi sama, mis kõige suurem vaatamiste arv, kuid vähim vaatamiste arv ning viimasesse punkti jõudnute arv enamasti kattuvad. Viimased ei ühtinud ilmselt juhtudel, kui oli esitatud murelahendajas mõni vihje, mida õppurid ei vajanud ning oli ilma abitagi suuremale osale üliõpilastest selge. Suurimat arvu kajastas enamasti mõni vahepunkt. Sellise punkti puhul ilmselt vajati kas rohkem abi ehk kasutati korduvalt või suurenes see number selle arvelt, et murelahendajate keskkonnas on pärast vastuse vaatamist vaja minna tagasi küsimuse juurde selleks, et saada järgmise sammu juurde.



Joonis 7. Kõige suurem ning väiksem murelahendajate vaatamiste arv



Joonis 8. Murelahendajatesse sisenemine, abi saamine ja lõppu jõudmine

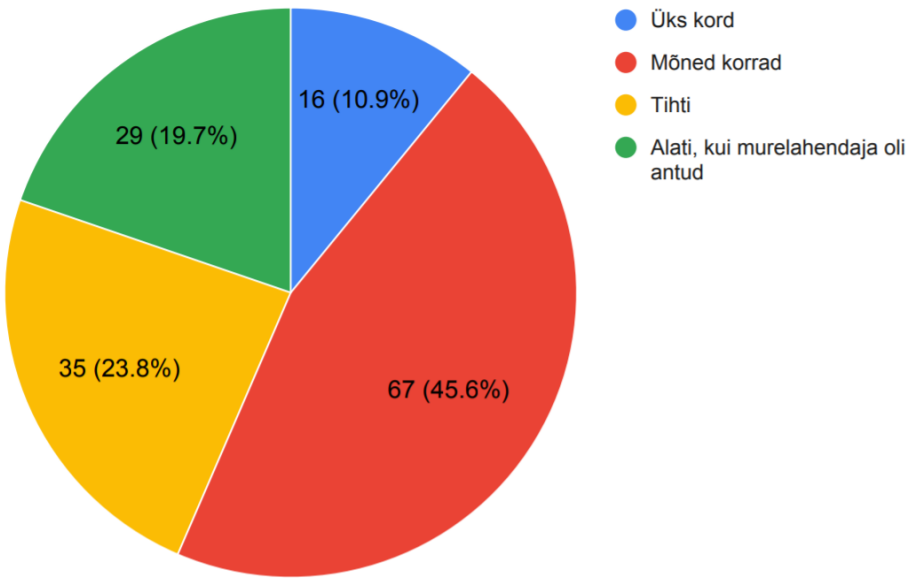
6. Tagasiside ja analüüs

6.1 Tagasisidestamine

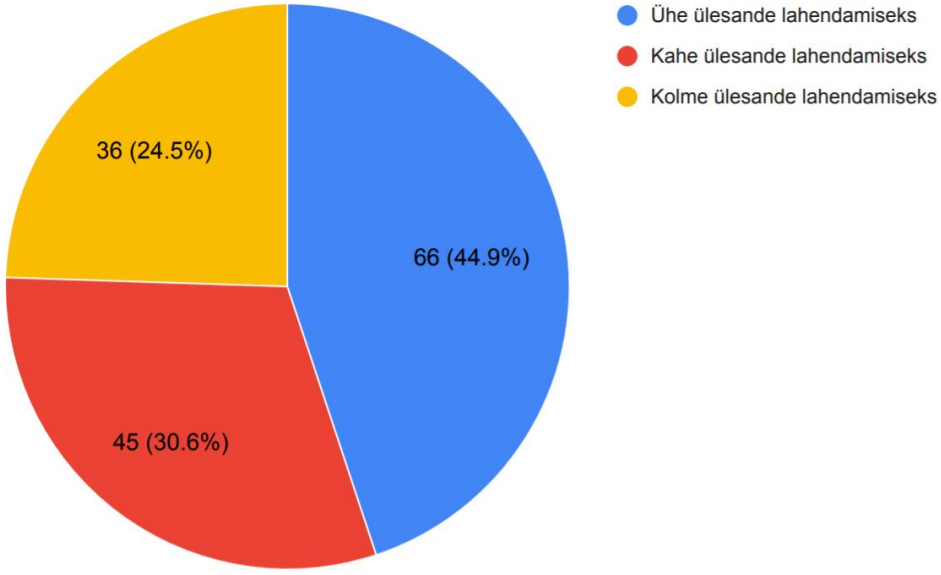
Üliõpilaste arvamuse teadasaamiseks murelahendajate kohta viidi kursusel osalenute vahel läbi tagasisideküsitlus (Lisa II), millele vastamine ei olnud kohustuslik, kuid täitmise eest oli võimalik teenida kursusel üks lisapunkt. Kokku andis tagasisidet 189 üliõpilast ehk ligikaudu 65% kursusel osalejatest. Küsitlus oli jagatud kaheks sektsiooniks. Esimeses osas tuli vastajatel avaldada punkti saamiseks oma nimi, kuid vastuste analüüsimisel oli tagatud täielik anonüümsus. Lisaks sellele oli küsitud, kas üliõpilane oli kursuse jooksul vähemalt korra murelahendajat kasutanud. Nende jaoks, kes vastasid viimasele küsimusele eitavalt, oli küsimustik lõppenud. Selliseid üliõpilasi oli vastanute seas 42 tükki ehk 22,2%. Üliõpilaste käest, kes olid murelahendajatega siiski tutvunud ning neid kodutööde lahendamisel kasutanud, küsiti veel täiendavat tagasisidet, seega edasisi vastajaid oli 147 ehk 77,8%. Teises alamjaotuses tuli vastata üheksale küsimusele. Küsimused olid nii ühe- kui ka mitme vastusevalikuvariandiga tüüpi, aga rakendati ka Likerti skaalat ning oli ka üks mittekohustuslik avatud vastusega küsimus. Näiteks uuriti üliõpilastelt kas ning milliste murelahendajate puhul nad abi said, kui tihti nad neid kasutasid või millises järjekorras nad kodutööde lahendamisele lähenesid. Avatud küsimuses oli võimalik teha ettepanekuid, kuidas murelahendajaid edasi arendada või mida nende puhul muuta.

6.2 Murelahendajate kasulikkus

Esmalt analüüsiti seda, kui palju kasutasid üliõpilased murelahendajate abi. 147-st vastanust ning vähemalt korra murelahendajaid kasutanud üliõpilastest peaaegu pooled kasutasid murelahendajaid vaid mõned korrad ehk võib arvata, et paljud neist siiski pidevat abi kursuse raames ei vajanud. Nagu jooniselt 9 näha, siis peaaegu teine pool kursusest ehk 43,5% vastanutest kasutas murelahendajate abi tihti või lausa alati, kui need olid olemas. Seega vastuste põhjal võib öelda, et ligikaudu pooled vajavad siiski pidevat abi või rohkem suunitlusi materjali paremaks omandamiseks. Teisest küljest saab jooniselt 10 välja lugeda, et ~45% vastanutest kasutas ühe praktikumi raames vaid ühe ülesande lahendamiseks murelahendaja abi.

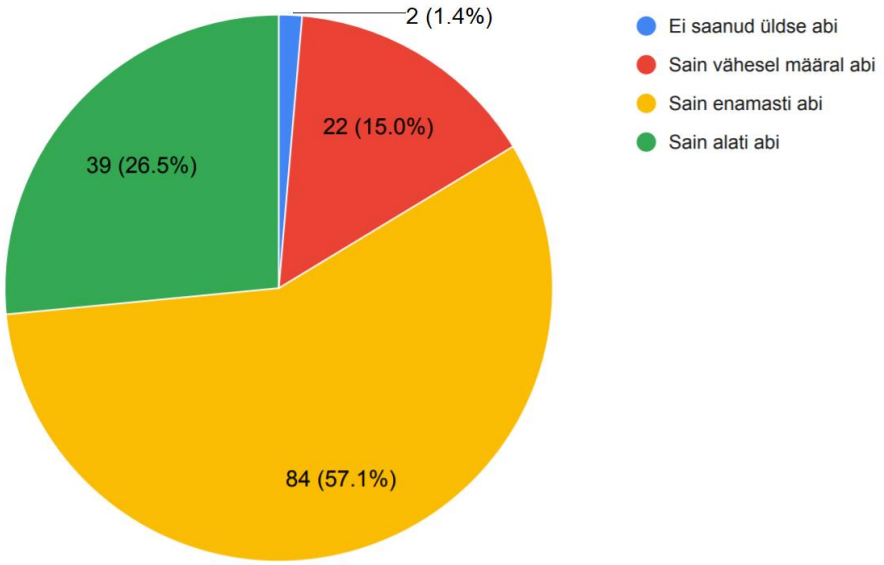


Joonis 9. Murelahendajate kasutamise sagedus



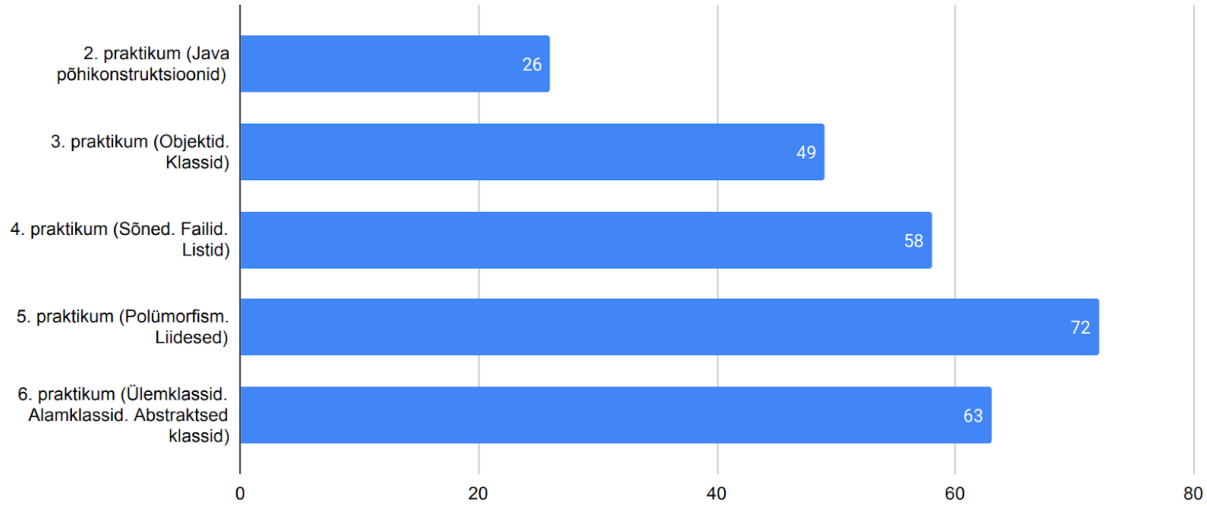
Joonis 10. Murelahendajate kasutamise arv ühe praktikumimaterjali lõikes

Kõige otsesemat kasulikkust üliõpilaste enda arvates saab hinnata ilmselt selle põhjal, millisel määral nad enda hinnangu kohaselt murelahendajatest kodutööde lahendamisel abi said. Tulemuste põhjal, mis kajastuvad joonisel 11, võib öelda, et murelahendajaid kasutanud õppurid peavad nimetatud abisüsteemi kasulikuks. Nimelt 83,6% vastanutest sai enamasti või alati abi ning ülejäänud 16,4% oleks murelahendajatest eeldanud rohkemat ehk said vähesel määral või ei saanud üldse abi.



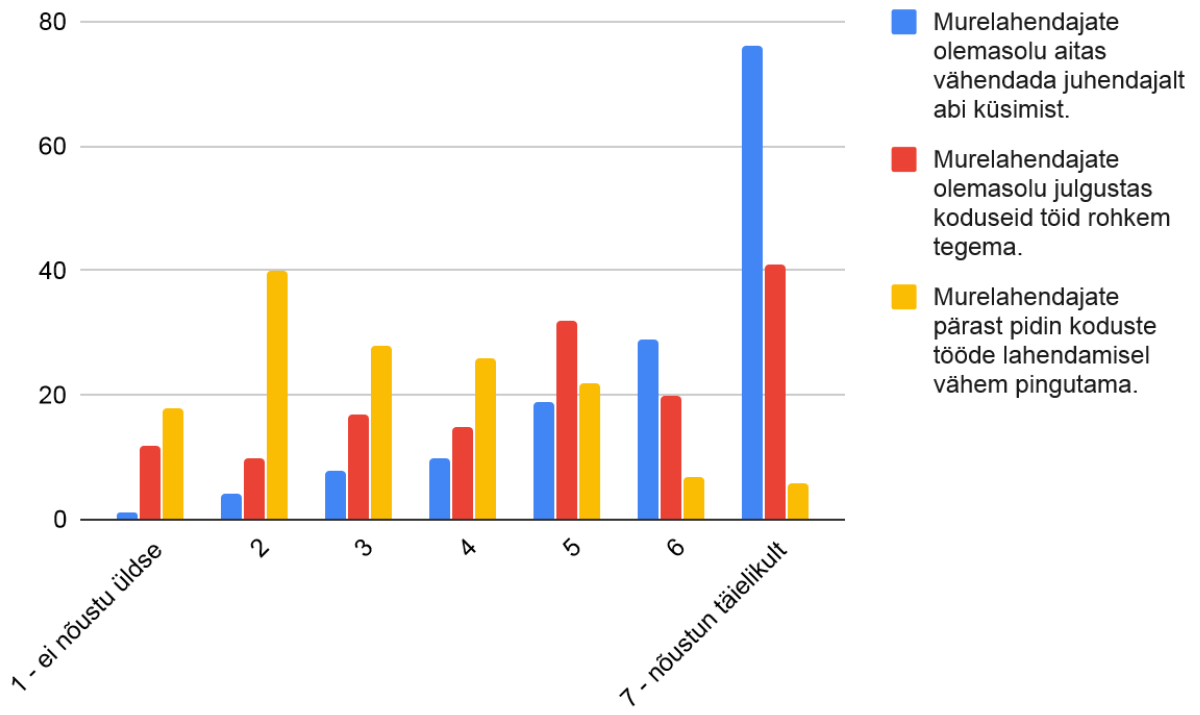
Joonis 11. Abi saamise määr murelahendajatest kodutööde lahendamisel

Veel küsiti üliõpilastelt, milliste praktikumide puhul said nad murelahendajatest kõige enam abi ning sealjuures oli võimalik valida ka mitu vastusevarianti. Kasumäär oli kasvavas trendis iga eelneva praktikumi suhtes ning kõige enam vajalikku informatsiooni kodutööde lahendamiseks said üliõpilased kursuse keskpaiku polümorfismi ja liideste teema korral (joonis 12). Mõndade vastanute arvates oli esimeste praktikumide puhul kõik hästi lahti seletatud, aga oli ilma nendetagi selge, kuid hilisemates praktikumides jäid vihjed puudulikuks. Teised seevastu tundsid, et esimesed murelahendajad aitasid hästi järje peale saada ning lõid tugeva baasi edaspidiseks. Lisati, et ka kursuse teise poole praktikumide tarbeks võiks abimaterjalid olemas olla, kuid praeguses kontseptsioonis on mõeldud murelahendajate abil kursuse esimesel poolel tugev aluspõhi luua, millele tuginedes üliõpilased ka edaspidi edukalt hakkama saaks.

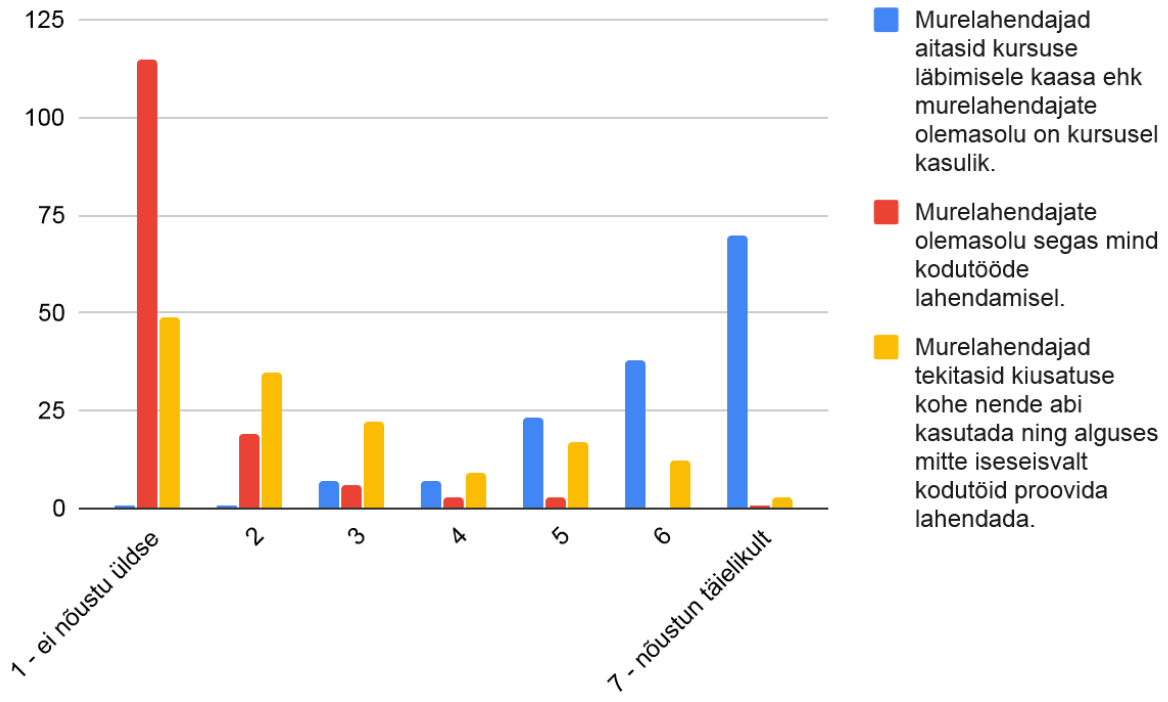


Joonis 12. Üliõpilaste hinnang murelahendajate kasule praktikumide lõikes

Veel esitati üliõpilastele erinevad väited, kuidas murelahendaja olemasolu kogu protsessi mõjutas. Neid tuli hinnata Likerti 7-palli skaala alusel, kus „1“ tähistab mittenõustumist ning „7” täielikku väitega nõustumist. Näiteks vastanute seisukohale tuginedes võib öelda, et täidetud sai ka eesmärk vähendada juhendajatele esitatavate küsimuste arvu. 76 vastanut nõustus absoluutselt väitega, et murelahendajate olemasolu aitas vähendada õppejõult abi küsimist ning vaid üks oli täiesti vastupidisel arvamusel (joonis 13). Murelahendajate olemasolu enamasti kodutööde lahendamisel ei seganud ehk tulemused olid arvukalt täieliku mittenõustumise poolel, kuid teisest küljest väitele „murelahendajate olemasolu tekitas kiusatust kohe nende abi kasutada” puhul olid vastused jagunenud palju ühtlasemalt kõikide vastusevariantide vahel (joonis 14).



Joonis 13. Murelahendajate mõju kodutööde lahendamisele I



Joonis 14. Murelahendajate mõju kodutööde lahendamisele II

6.3 Murelahendajate roll õppeprotsessis

Käesoleva töö raames sooviti välja selgitada veel seda, kas murelahendajad võivad mõjuda pärssivalt üliõpilaste õpingutele ehk näiteks pöördutakse murelahendajate poole juba ilma esmalt ise ülesannet lahendamata. Näiteks uuriti üliõpilastelt, millises järjekorras nad koduste tööde lahendamisele lähenesid. Järjestada tuli järgmised tegevused: ülesannete lahendamine, murelahendajate kasutamine abi saamiseks, sõbra poole pöördumine abi saamiseks, praktikumijuhendaja poole pöördumine abi saamiseks, muu (näiteks internetist abi otsimine). Kui tegevus protsessi ei kuulunud, tuli see märkida 0-ga. 135 õppurit ehk üle 90% vastas, et esmalt asuti ülesandeid lahendama ning alles seejärel otsiti vajadusel ka abi. Oleks võinud arvata, et sõprade või praktikumijuhendajate poole pöördumine on populaarsem, kuid ligikaudu sadakond üliõpilast vastas, et ei teinud seda üldse. Murelahendajate kasutamine teise sammuna oli aga märgitud 61 vastajal, samas sellest populaarsemaks osutus siiski muu ehk näiteks internetist abi otsimine, mille märkis number kahega 75 üliõpilast. Selle põhjuseks võib olla see, et internetist saab otsida vastust kindlale küsimusele, kuid murelahendajate puhul tuleb käia kogu puu läbi ning probleemiks võib olla näiteks mingi veateade, mida praegusel juhul murelahendajates üldse ei käsitleta. Sagedamini esinevate veateadete samuti vaatluse alla võtmine oli ka üks õpilastepoolseid soovitusi murelahendajate edasiarenduseks, mis ehk aitaks tõsta murelahendajate eelistust muudele abivahenditele ehk näiteks internetile. Kõige populaarsem oli murelahendajate kasutamine teise või kolmanda sammuna, vastavalt 61 ning

52 vastust. Kohe esimese valikuna pöördus murelahendajate poole vaid 4 õppurit ehk võib arvata, et murelahendajate olemasolu ei „kuritarvitata".

6.4 Avatud tagasiside

Küsimustikus oli ka üks avatud lahter, kus oli võimalus anda soovitusi edasiarenduseks ning ka konstruktiivset kriitikat. Vastustes esines olulisi lahkhelisid ning nagu ka korduvalt varasemalt mainitud, siis nii suure grupi jaoks ühte toimivat lahendust välja töötada ei olegi praktiliselt võimalik. Ühest küljest sooviti näha rohkem reaalset koodi, teised soovisid aga rohkem üldise pildi lahti seletamist ning arvasid et koodi ette andmine loob võimaluse selle plagieerimiseks. Mõlema osapoole tarbeks käidi välja ka kasulikud ideed. Näiteks laiema tausta mõistmiseks lisada viiteid dokumentatsioonidele ja loengu- või muudele lisamaterjalidele, kust õppur saab soovi korral iseseisvalt juurde õppida. Koodinäidete puhul aga esitada mingi analoogse ülesande lahenduskäik samm-sammult. Vastastikused arvamused esinesid ka etteantavate vihjete sisukuse kohta. Toodi välja, et hilisemates ehk keerulisemates praktikumides võiks olla sisu rohkem lahti seletatud, teisalt tõsteti korduvalt esile, et esimestes praktikumides üsnagi üheselt mõistetavad vihjed aitasid luua selgema pildi kogu kursusest. Kohati tekitasid napisõnalised vihjed kasutajates isegi rohkem segadust kui selgust. Korduvalt toodi välja ka olukordi, kus kodutöö lahendamine ei jäänud oskuste, vaid ebaselgete juhiste taha kinni. Seda saaks lahendada kahel viisil: teha muudatusi praktikumimaterjalide ülesannete sõnastustes või kirjutada murelahendajates ülesande tekst teiste sõnadega rohkem lahti. Üldises pildis said murelahendajad siiski väga positiivset tagasisidet. Näiteks mainiti korduvalt ka seda, et isegi kui saadi ülesandega iseseisvalt hakkama, siis käidi silmaringi laiendamise ning koodi efektiivsemaks muutmise mõttes murelahendajad veel lisaks läbi.

6.5 Tagasiside murelahendajate keskkonnale

Olgugi, et murelahendajate keskkond ei olnud käesoleva lõputöö osa, siis seegi oli osa programmi kasutajakogemusest üliõpilaste jaoks ehk konstruktiivset tagasisidet anti ka sellele. Üheks välja käidud ideeks oli vahetu tagasisidestamine ehk võimalus hinnata käigu pealt kindlat plokki. Kui murelahendajate haldaja näeb, et millelegi kindlale on tulnud palju negatiivset tagasisidet, siis saab seda vajadusel kohe muuta. Positiivse puhul saab aga aimu, milline esitusviis on üliõpilastele sobivaim. Mitmeid soovitusi tuli ka programmi kasutajamugavuse ning -liidese kohta. Näiteks võiks olla lisatud nupp, mis viib kohe algusesse tagasi või vähendada „tagasi" nupu pidevat kasutamise vajadust ehk toodi välja, et võrdlemisi keeruline on erinevate vihjete vahel liikumine. Ka nuppude tekstid ei olnud murelahendajate kasutajate

jaoks igas olukorras üks-üheselt mõistetavad, kuid õnneks on programmil funktsionaalsus olemas, mis lubab nuppude tekste murelahendajate haldajal muuta.

Autoril tekkis samuti mõnigaid probleeme murelahendajate vormistamisega keskkonnas, kui näiteks mõni nupp ilmus topelt ning selle eemaldamiseks tuli kogu samm ära kustutada ning uuesti sisestada. Ilmselt on koodis mingi sisuline viga, mis vajab tulevikus parandamist. Lisaks sellele võiks lugeda vahelehe sulgemist ajal kui murelahendajaga pole lõpuni jõutud kui abi saamist, sest niimoodi saaks koguda täpsemat statistikat murelahendaja kasulikkuse kohta.

7. Kokkuvõte

Käesoleva bakalaureusetöö eesmärk oli luua kursuse „Objektorienteeritud programmeerimine" tarbeks murelahendajad, mis aitaksid üliõpilasi vastava programmeerimisstiili loogika omandamisel ning kursusel nõutud kodutööde lahendamisel. Töö raames töötati välja murelahendajad 5 praktikumi jaoks ehk kokku 15 murelahendajat. Koostatud abimaterjalidele üliõpilastelt saadud tagasiside põhjal sooviti välja selgitada, kas murelahendajate süsteem on tarvilik lisafunktsionaalsus kursusele „Objektorienteeritud programmeerimine".

Töös anti ülevaade objektorienteeritud programmeerimisest ning Tartu Ülikoolis selle õpetamisel kasutatavast programmeerimiskeelest Java. Veel uuriti pööratud õppe vormi olemust, selle negatiivseid ja positiivseid külgi ning rakendamist erinevate kõrgkoolide näitel. Lisaks tutvustati töös Tartu Ülikooli „Objektorienteeritud programmeerimine" kursust, mille tarbeks murelahendajad loodi.

Koostatud murelahendajate kohta viidi üliõpilaste seas läbi tagasisideküsitlus, saamaks teada, kas üliõpilased pidasid vastavaid abimaterjale neile kasulikuks. Küsitlusele vastas 189 üliõpilast, kellest 147 ehk 77,8% oli kursuse jooksul kasutanud murelahendajaid vähemalt korra. Tulemustest selgus, et 83,6% murelahendajaid kasutanud vastanutest said nendest ka enamasti või alati abi. Lisaks sellele ilmnes, et peaaegu alati pöörduti murelahendajate poole alles siis, kui vajati abi, mitte ei asutud neid kasutama esimese sammuna kodutöö lahendamise protsessis. Murelahendajaid kasutati ka juhul, kui kodutöö lahendati iseseisvalt, kuid sooviti muuta oma koodi efektiivsemaks ning saada täiendavaid soovitusi. Üliõpilaste sõnul aitasid murelahendajad vähendada ka praktikumijuhendajatelt abi küsimist, mis omakorda aitab vähendada õppejõudude ning -assistentide lisa töökoormust. Üldine tagasiside murelahendajatele oli positiivne ning üliõpilased pidasid neid kasulikeks abimaterjalideks, seega tasub neid rakendada ka edaspidistel aastatel. Edasiarendusena tasub olemasolevaid murelahendajaid vajadusel modifitseerida ning üliõpilaste tagasisidet arvesse võttes luua murelahendajad ka kursuse teise poole praktikumide tarbeks.

8. Viidatud kirjandus

[1] Tartu Ülikooli õppeinfosüsteem. https://ois2.ut.ee (04.04.2020)

[2] Boustedt, J., Eckerdal, A., Mccartney, R., Moström, J. E., Ratcliffe, M., Sanders, K., Zander, C. Threshold Concepts in Computer Science: Do they exist and are they useful? *ACM SIGCSE Bulletin*, pp. 504-508, 2007.

[3] Lepp, M. Objektorienteeritud programmeerimine. Loenguslaidid, 2020. https://courses.cs.ut.ee/LTAT.03.003/2020_spring/uploads/Main/oop20kevadloeng1teg.pdf (21.04.2020)

[4] Tan, P.-H., Ting, C.-Y., Ling, S.-W. Learning Difficulties in Programming Courses: Undergraduates' Perspective and Perception. *International Conference on Computer Technology and Development*, pp. 42-46, 2009.

[5] Lawan, A. A., Abdi, A. S., Abuhassan, A. A., Khalid, M. S. What is Difficult in Learning Programming Language Based on Problem-Solving Skills? *International Conference on Advanced Science and Engineering (ICOASE)*, pp. 18-22, 2019.

[6] Lepp, M., Palts, T., Luik, P., Papli, K., Suviste, R., Säde, M., Hollo, K., Vaherpuu, V., Tõnisson, E. Troubleshooters for Tasks of Introductory Programming MOOCs. *International Review of Research in Open and Distributed Learning*, Vol. 19, No. 4, 2018.

[7] Blansit, B. D. Object Oriented Programming: What is IT Talking About? *Journal of Electronic Resources in Medical Libraries*, Vol. 7, pp. 90-97, 2010.

[8] Oracle kodulehekülg. Lesson 8: Object-Oriented Programming. https://www.oracle.com/technetwork/java/oo-140949.html (12.12.2019)

[9] Eesti Keele Instituudi võõrsõnade leksikon. http://www.eki.ee/dict/vsl/index.cgi (14.12.2019)

[10] Oracle Java dokumentatsioon. Lesson: Object-Oriented Programming Concepts. https://docs.oracle.com/javase/tutorial/java/concepts/index.html (23.03.2020)

[11] Jian, S., Wenyong, W., Zebing, W. A Teaching Path for Java Object Oriented Programming. *International Forum on Information Technology and Application*, pp. 465-468, 2009.

[12] Oracle kodulehekülg. The History of Java Technology. https://www.oracle.com/java/technologies/javahistory.html (30.03.2020)

[13] Tiobe kodulehekülg. Tiobe Index. https://www.tiobe.com/tiobe-index/ (30.03.2020)

[14] cv.ee kodulehekülg. Tööpakkumised. https://www.cv.ee/toopakkumised/infotehnoloogia/q-java?sort=inserted&dir=desc (30.03.2020)

[15] Liang, Y. D. Characteristics of Java (Optional). https://docplayer.net/15119984-Characteristics-of-java-optional-y-daniel-liang-supplement-for-introduction-to-java-programming.html (30.03.2020)

[16] Lepp, M., Tõnisson, E. Integrating Flipped Classroom Approach and Work in Pairs into Workshops in Programming Course. *International Conference on Frontiers in Education: Computer Science and Computer Engineering*, pp. 220-226, 2015.

[17] Gannod, G. C., Burge, J. E., Helmick, M. T. Using the inverted classroom to teach software engineering. *ACM/IEEE 30th International Conference on Software Engineering*, pp. 777-786, 2008.

[18] Manoj Kumar P., Renumol V.G., Murthy, S. Flipped Classroom Strategy to Help Underachievers in Java Programming. *International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*, pp. 44-49, 2018.

[19] Chis, A. E., Moldovan, A.-N., Murphy, L., Pathak, P., Muntean, C. H. Investigating Flipped Classroom and Problem-based Learning in a Programming Module for Computing Conversion Course. *Journal of Educational Technology & Society*, Vol. 21, No. 4, pp. 232-247, 2018.

[20] Alhazbi, S. Using Flipped Classroom Approach to Teach Computer Programming. *IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 441-444, 2016.

[21] Cambridge Dictionary. https://dictionary.cambridge.org/ (10.04.2020)

[22] Jonassen, D. H., Hung, W. Learning to Troubleshoot: A New Theory-Based Design Architecture. *Educational Psychology Review*, Vol. 18, No. 1, pp. 77-114, 2006.

[23] Vaherpuu, V. Murelahendajate loomise keskkond. Tartu Ülikooli arvutiteaduse instituudi bakalaureusetöö, 2016. https://comserv.cs.ut.ee/ati_thesis/datasheet.php?id=53433&year=2016 (24.04.2020)

Lisad

I. Koostatud murelahendajad

Praktikum 2. Ülesanne 4. : https://progtugi.cs.ut.ee/#/ts/5e42775f50ad8d0325d298b4/

Praktikum 2. Ülesanne 5. : https://progtugi.cs.ut.ee/#/ts/5e414e55b90f476b1b4175fa/

Praktikum 2. Ülesanne 7. : https://progtugi.cs.ut.ee/#/ts/5e41772050ad8d0325d297e5/

Praktikum 3. Ülesanne 3. : https://progtugi.cs.ut.ee/#/ts/5e4a58f750ad8d0325d2abef/

Praktikum 3. Ülesanne 4. : https://progtugi.cs.ut.ee/#/ts/5e4a621150ad8d0325d2ac31/

Praktikum 3. Ülesanne 6. : https://progtugi.cs.ut.ee/#/ts/5e4a652550ad8d0325d2ac5b/

Praktikum 4. Ülesanne 2. : https://progtugi.cs.ut.ee/#/ts/5e4f9e4e50ad8d0325d2db1c/

Praktikum 4. Ülesanne 7. : https://progtugi.cs.ut.ee/#/ts/5e4fa21b50ad8d0325d2db4d/

Praktikum 4. Ülesanne 8. : https://progtugi.cs.ut.ee/#/ts/5e4faa1f50ad8d0325d2dba2/

Praktikum 5. Ülesanne 2. : https://progtugi.cs.ut.ee/#/ts/5e57e19150ad8d0325d318b7/

Praktikum 5. Ülesanne 3. : https://progtugi.cs.ut.ee/#/ts/5e57e76750ad8d0325d31923/

Praktikum 5. Ülesanne 4. : https://progtugi.cs.ut.ee/#/ts/5e57f7d350ad8d0325d319df/

Praktikum 6. Ülesanne 4. : https://progtugi.cs.ut.ee/#/ts/5e661f5c50ad8d0325d37812/

Praktikum 6. Ülesanne 5. : https://progtugi.cs.ut.ee/#/ts/5e6623c150ad8d0325d37863/

Praktikum 6. Ülesanne 6. : https://progtugi.cs.ut.ee/#/ts/5e6641f750ad8d0325d37a6a/

II. Tagasisideküsimustik

Sektsioon I

Tagasisideküsitlus murelahendajate kohta

Tere! Olen Tartu Ülikooli informaatika eriala kolmanda aasta tudeng Anett Klaanberg ning minu lõputööks oli luua aine "Objektorienteeritud programmeerimine" tarbeks murelahendajad ehk kodutööde lahendamist abistavate vihjete komplektid.
Kuna sedalaadi abivahendite formaati rakendatakse kursuse raames esmakordselt, siis on tarvis sellele ka tagasisidet ja kes oleks veel paremad tagasisidet andma kui mitte murelahendajate kasutajad. Et saaksime edaspidi veel paremaid abimaterjale kursustele luua, siis ole hea ja anna murelahendajate kohta tagasisidet. Mida ausamalt ja detailsemalt, seda parem! :)
Suur-suur aitäh!

*Küsitluses kogutud informatsiooni puhul on tagatud täielik anonüümsus!

Eesnimi (oma nimi on vaja esitada juhul, kui soovid kursuse raames ühte lisapunkti teenida)

Your answer

Perekonnanimi (Oma nimi on vaja esitada juhul, kui soovid kursuse raames ühte lisapunkti teenida. Vastustuste puhul on tagatud täielik anonüümsus)

Your answer

Kas kasutasid kursuse "Objektorienteeritud programmeerimine" raames kodutööde lahendamisel murelahendajate abi? *

- ◯ Jah
- ◯ Ei

Sektsioon II

Kui tihti kasutasid murelahendajate abi? *

- ◯ Üks kord
- ◯ Mõned korrad
- ◯ Tihti
- ◯ Alati, kui murelahendaja oli antud

Mitme murelahendaja abi kasutasid keskmiselt ühe praktikumi kodutööde lahendamiseks? (Iga ülesande kohta oli üks murelahendaja) Kasutasin enamasti murelahendaja abi ... *

- ◯ ühe ülesande lahendamiseks
- ◯ kahe ülesande lahendamiseks
- ◯ kõikide ülesannete lahendamiseks

Millisel määral said murelahendajatest abi kodutööde lahendamisel? *

- ◯ Ei saanud üldse abi
- ◯ Sain vähesel määral abi
- ◯ Sain enamasti abi
- ◯ Sain alati abi

Vasta, millises ulatuses nõustud järgnevate väidetega. *

	1 - ei nõustu üldse	2	3	4	5	6	7 - nõustun täielikult
Murelahendajate olemasolu aitas vähendada juhendajalt abi küsimist.	○	○	○	○	○	○	○
Murelahendajate olemasolu julgustas koduseid töid rohkem tegema.	○	○	○	○	○	○	○
Murelahendajate pärast pidin koduste tööde lahendamisel vähem pingutama.	○	○	○	○	○	○	○
Murelahendajad aitasid kursuse läbimisele kaasa ehk murelahendajate olemasolu on kursusel kasulik.	○	○	○	○	○	○	○
Murelahendajate olemasolu segas mind kodutööde lahendamisel.	○	○	○	○	○	○	○
Murelahendajad tekitasid kiusatuse kohe nende abi kasutada ning alguses mitte iseseisvalt kodutöid proovida lahendada.	○	○	○	○	○	○	○

Millises järjekorras lähenesid kodutööde lahendamisele? Esimene tegevus märkige numbriga 1, teisena alustatud tegevus numbriga 2, kolmas tegevus numbriga 3 jne. Tegevus, mida üldse ei teinud, märkige nulliga (selliseid võib ka mitu olla). *

	0	1	2	3	4	5
Ülesannete lahendamine	○	○	○	○	○	○
Murelahendajate kasutamine abi saamiseks	○	○	○	○	○	○
Sõbra poole pöördumine abi saamiseks	○	○	○	○	○	○
Praktikumijuhendaja poole pöördumine abi saamiseks	○	○	○	○	○	○
Muu (näiteks internetist abi otsimine)	○	○	○	○	○	○

Kas murelahendajas ette antud vihjed olid piisavad? *

- ○ Ei, liiga vähe oli abi antud
- ○ Jah, vihjed olid piisavad
- ○ Jah, aga isegi liiga palju oli ette antud

Millise praktikumi puhul oli murelahendajatest kõige enam kasu? (Valida saad ka mitu) *

- ☐ 2. praktikum (Java põhikonstruktsioonid)
- ☐ 3. praktikum (Objektid. Klassid)
- ☐ 4. praktikum (Sõned. Failid. Listid)
- ☐ 5. praktikum (Polümorfism. Liidesed)
- ☐ 6. praktikum (Ülemklassid. Alamklassid. Abstraktsed klassid)

Kas oled murelahendajate või millegi analoogsega ka varem kokku puutunud? *

- ○ Jah, ATI M00C'idel (Programmeerimisest maalähedaselt, Programmeerimise alused, Programmeerimise alused II)
- ○ Jah, ülikooli kursusel (Programmeerimise alused, Programmeerimise alused II)
- ○ Jah, kuskil mujal
- ○ Ei

On Sul ettepanekuid, kuidas edaspidi murelahendajaid paremini koostada või mida muuta?

Your answer

III. Litsents

Mina, **Anett Klaanberg**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose **Murelahendajate koostamine Tartu Ülikooli kursuse „Objektorienteeritud programmeerimine" tarbeks**

mille juhendaja on Marina Lepp,

reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Anett Klaanberg

08.05.2020