

**Projekt “JOBIT - Innovative teaching methodologies and courseware for software development VET to reduce skills gap in IT”**

**Käsiraamat noorem tarkvaraarendaja eriala kutseõpetajale**

Viimati uuendatud: Juuni 2017

Koostajad: European University Cyprus (Cyprus)

BCS Koolitus (Estonia)

Centro Internazionale per la Promozione dell’ Educazione e lo Sviluppo (Italy)



This project has been funded with support from the European Commission.

This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Sisukord

[Sissejuhatus 3](#_Toc491701993)

[Mis on JOBIT? 4](#_Toc491701994)

[Tänusõnad 5](#_Toc491701995)

[Kutsepedagoogika mõiste 6](#_Toc491701996)

[Õpiväljundid kutseõppes 7](#_Toc491701997)

[Õpetatavad ained 8](#_Toc491701998)

[Kutseõppele sobivad õppimis- ja õpetamismeetodid 9](#_Toc491701999)

[Kuidas seda kõike rakendada nooremtarkvaraarendaja õppeainetes? 10](#_Toc491702000)

[Scrum 13](#_Toc491702001)

[Metoodika 13](#_Toc491702002)

[Scrumi rollid 13](#_Toc491702003)

[Scrum hariduses 14](#_Toc491702004)

[Kuidas kasutada Scrumi kutseõppes? 16](#_Toc491702005)

[Kasutatud allikad 18](#_Toc491702006)

[LISA 1 – Tunnikavade näidised 19](#_Toc491702007)

[1. Tunnikava - Programmeerimine 19](#_Toc491702008)

[2. Tunnikava näidis – Tarkvaraarendus 24](#_Toc491702009)

# Sissejuhatus

Antud käsiraamat on koostatud noorem tarkvaraarendaja (*Junior Software Development* - JSD) eriala õpetajate koolitamiseks – **mida ja kuidas õpetada**, etõpilased omandaksid paremini olulisi õpiväljundeid. Pilootkoolitused õpetajatele toimusid mais‑juunis 2017 ja osalejatelt saadud tagasiside põhjal on käsiraamatut jooksvalt täiendatud.

Koolituse läbinu (oodatavad õpiväljundid):

* Mõistab kutseõppe olemust, tunneb õpetamisstrateegiaid ja -meetodeid kutseõppe õpilaste õpetamiseks, saab aru kutseõppe erisustest võrreldes üldharidusega.
* Teab, kuidas kujundada kutseõpetaja identiteeti (tihedalt seotud nende pedagoogilise lähenemisega; sotsiaalpoliitiline mõju; inimlik ja eetiline dimensioon (Tran & Nguyen, 2013)).
* Mõistab paremini, kuidas koostada tarkvaraarenduse eriala õpiväljundeid, arvestades kutseõppe praeguseid trende ja vajadusi ning töömaailma ootusi.
* Õpib kasutame projekti käigus loodud õppematerjale.
* Suudab genereerida ja arendada ideid tööandjate kaasamiseks õppetöösse, oskab leida kontakte töömaailmaga, et olla kursis töömaailma arengutega ja trendidega.
* Tutvub tarkvaraarenduse töövahendite ja pedagoogiliste võtetega, mis sobivad kõige paremini kutseõppesse.
* Kasutab tarkvaraarenduse ainete õpetamisel sobivaid meetodeid, lähtub projekti õppematerjalides toodud soovitustest, arendab ja levitab neid.

# Mis on JOBIT?

Tarkvaraarendus laiemas mõttes on muutumas üha olulisemaks valdkonnaks tänases maailmas ning see on seotud väga erinevate erialadega. On üldtuntud fakt, et enamikus Euroopa riikides tuntakse kõige enam puudust tarkvaraarendajatest. Erinevad allikate põhjal võib väita, et töömaailm vajab tuhandeid või isegi kümneid tuhandeid uusi arendajaid.

Kuni viimaste aastateni oli enamikus Euroopa riikides arendajate haridusnõudeks magistri- või minimaalselt bakalaureuse kraadi olemasolu. Kuid tänasel päeval on olemas palju näiteid, kus programmeerimist õpetatakse edukalt juba põhikoolis, samuti on erinevates Euroopa riikides olemas EQF 3, 4 ja 5 taseme õppekavad.

Siiski peab märkima, et kutseõpe pole väga populaarne ja tööandajad kõhklevad, kas võtta tööle kutseõppe lõpetajaid, sest pole kindlad, kas kutseõpe ikka annab oskused, mida töömaailm täna tavapäraselt arendajalt eeldab.

Projekti „*Innovative teaching methodologies and courseware for the software development VET to reduce skills gap in Information Technology*“ (lühend JOBIT) fookuses on selle puudujäägi vähendamine arendades innovatiivseid tarkvaraarenduse õpetamise metoodikaid kutseõppele ja koolitusfirmadele EQF tasemetel 3-5. Projekti lõpuks on tarkvaraarendust õpetavad õpetajad varustatud oskustega erialaainete kvaliteetseks õpetamiseks ja teavad rohkem, mida töömaailm nende õpilastelt ootab.

Projekti eesmärk on suurendada noorem tarkavaraarendaja õppekava erialaainete sisu vastavust reaalsetele töömaailma vajadustele. See peaks tagama, et lõpetajad on tööturul edukamad ja nende tööhõive suureneb.

**Innovative teaching methodologies and courseware for the software development VET to reduce skills gap in Information Technology**

Projekti JOBIT partnerid:

* BCS Koolitus (Estonia)– www.bcskoolitus.ee
* CEIPES - Centro Internazionale per la Promozione dell’Educazione e lo Sviluppo (Italy)– www.ceipes.org
* European University Cyprus (Cyprus)– www.euc.ac.cy.

Project JOBIT tegevused ja sihtgrupid:

* Uuring partnerriikides, millised on tööandjate ootused nooremtarkvaraarendaja eriala lõpetajate teadmistele ja oskustele.
* Leida sobivaid ja arendada metoodikaid nooremtarkvaraarendaja eriala olulisi ainete õpetamiseks.
* Pakkuda õppematerjale, mis on seotud projekti käigus loodud õppekavade, programmide, töövahendite ja metoodikatega.
* Kõikides partnerriikides toimuvad õpetajate koolitused – klassikoolitused ja ettevõtte praktika, mille tulemusena tõuseb õpetajate erialane kompetentsus.

# Tänusõnad

Et projekt oleks edukas, siis alustuseks määratles konsortsium projekti eesmärgid - vajaduse suurendada kutseõppe vastavust töömaailma ootustele st tarkvaraarenduse eriala lõpetajate oskused oleksid piisavad alustamaks tööd nooremarendaja või -testijana.

Eesmärgi saavutamiseks tegid kõik partnerid uuringu erinevate huvigruppide seas. Eestis saadeti küsitlus kõikidele tarkvaraarenduse eriala õpetatavatele koolidele (kutseõpetajatele). Täname kõiki õpetajaid, kes uuringus osalesid ja põhjalikult vastasid. Uuringu tulemustest selgusid programmeerimiskeeled, töövahendid ja muud teemad, mida koolitustel käsitleda.

Eriline tänu kuulub väikesele Eesti tarkvarafirmale I-Sepp OÜ, mille juht Üllar Närep oli lahkesti nõus koostööd tegema ja ettevõtte töötajad aitasid koostada koolituse põhilisi õppematerjale.

Veel soovime tänada kõiki, kes osalesid koolituste ettevalmistamisel ja koolitajaid:

Kadri Vahtramäe, Kristen Gilden, Amir Zare Pashaei ja Helena Jeret-Mäe.

Suur tänu nendele ettevõtetele, kes olid nõus vastu võtma ja tutvustama oma ettevõtet, tüüpilist tarkvaraarendusprotsessi ja kasutatavaid vahendeid. Kõik ettevõtete külastused olid õpetlikud ja kasulikud. Eestis olid lahked võõrustajad:

1. SMIT - Annika Sepp, Merje Laasi, Evelin Trink, Kristine Baumann, Einar Koltšanov
2. Elisa - Villu Teearu, Martin Avik
3. Helmes - Raul Ennus ja Markus Karileet
4. Nortal - Külli Raidma ja Margus Hanni

# Kutsepedagoogika mõiste

**Võimatu on selgitada kutsepedagoogika mõistet või teoreetilist käsitlust ilma et sa ei ole esitanud ja leidnud vastust paarile kutseõppe fundamentaalsele küsimusele.**

Kutsepedagoogika on pedagoogika haru, mis hõlmab nii kutseõppedidaktikat kui ka ‑kasvatust. Kasutades mõistet “kutseõppe pedagoogika”, “kutsepedagoogika” mõistame seda kui “kutseõppes õpetamise ja õppimise teadust, kunsti ja meisterlikkust” (“What it is, why it matters and how to put it into practice“, Lucas, 2014: 2).

Kokkuvõtlikult on kutsepedagoogika õpetamisteadus, mis vastab küsimustele MIDA? ja KUIDAS? õpetada ning käsitleb seega õppesisu ja õppemeetodite küsimusi. Püüdes veelgi täpsemalt defineerida, siis kutsepedagoogika on arusaamade ja võtete kogum, kuidas kutseõpetaja igapäevaselt õpetab, kuidas kohandab oma õpetamismeetodikat, et see vastaks õpilaste igavesti muutuvatele vajadustele ja sobituks kontekstiga, milles nad ennast leida võivad (Hansen, 2008).

Kutsepedagoogika on silmnähtavalt vähe uuritud ja vähese teoreetilise käsitlusega valdkond (Rojewski, 2009; Lucas, 2014). Kutseõpe on väga sageli akadeemilise hariduse “vaene sugulane”. Kutsepedagoogika mõiste selgitamine on oluline, sest see sunnib meid mõtlema kutseõppe eesmärkidest laiemalt ja nendest arusaamine peaks aitama parandama ka kutseõppe mainet.

Kutsepedagoogika võimaldab meil arendada mudeleid ja vahendeid, mis aitaksid kutseõpetajat efektiivsemalt sobitada õpetamis- ja õppimismeetodeid õpilaste vajaduste ja kontekstiga. Sellises tähenduses mõjutab kutsepedagoogika otseselt õpetamise ja õppimise kvaliteeti. Lucas (2014) on kirjeldanud kutsepedagoogika teoreetilised põhitõed, mis tuleks selgeks teha, et mõista kutsepedagoogika olemust. Need põhitõed on esitatud järgneval joonisel

# Õpiväljundid kutseõppes

Kompetentsel töötajal (kutseõppe lõpetajal) peab olema palju erinevaid oskusi. Lucas (2013) on kaasajastanud ja laiendanud oskuste komplekti järgnevalt:

# Õpetatavad ained

Arvestades praeguseid kutseõppe trende ja vajadusi, tuleks nooremtarkvaraarendaja õppekaval õpetada järgmisi aineid. Õppeained põhinevad ka projekti esimesel etapil tehtud uuringu tulemustel (JOBIT- Output 1 – A3). Järgnev tabel näitab eelmises peatükis toodud oodatavate õpiväljundite ja õppeainete seoseid.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Õppeaine | Oodatavad õpiväljundid | | | | | |
|  | Asjatundlikkus | | | | | |
|  |  | Leidlikkus | | | | |
|  |  |  | Meisterlikkus | | | |
|  |  |  |  | Funktsionaalne kirjaoskus | | |
|  |  |  |  |  | Ettevõtlusega seotud hoiakud | |
|  |  |  |  |  |  | Üldoskused |
| Sissejuhatus informaatikasse | X | X | X |  |  |  |
| Programmeerimise alused 1 | X | X | X | X |  |  |
| Programmeerimise alused 2 | X | X | X | X |  |  |
| Veebiprogrammeerimine |  | X | X | X |  | X |
| Andmebaasid |  |  | X | X | X | X |
| Tarkvaraarendus 1 |  |  | X | X | X | X |
| Tarkvaraarendus 2 |  |  | X | X | X | X |
| Praktika ettevõttes (1 või 2 semestrit) | X | X | X | X | X | X |

Tabel 1.

# Kutseõppele sobivad õppimis- ja õpetamismeetodid

**Konstruktivism**

**Õppimine kui …**

Konstruktivism paneb rõhu õppimise protsessile, mitte õpiväljunditele. Õppimistegevused ja -keskkond peavad olema korraldatud nii, et õppijad saavad luua ja juhtida oma õppimise edenemist. Selles vaates on õpetaja ja treeneri ülesanded rohkem juhendamine ja treenimine kui õpetamine.

**Situatsiooniõpe (*Situated Learning*)**

Ettevõttesse praktikale minek peaks olema kavandatud nii, et üleminek teoreetiliselt õppelt formaalses klassiruumis teadmiste rakendamisele reaalses töökeskkonnas, oleks võimalikult sujuv.

**Autentsed ülesanded**

Spetsiaalselt koostatud ülesanded, mis rakendavad tavapäraseid teadmisi koos oskustega tulla toime reaalse elu väljakutsetega. Billett (2013) on defineerinud 4 põhimõtet, mis peaksid tagama kutseoskuste omandamise läbi autentese kogemuse:

1. Seotus tööülesandega (*engagement*)
2. Õppija vahetu juhendamine töökohal
3. Praktika töökohal
4. Õppija lähijuhendamine teiste töötajate ja ekspertide poolt.

**Õppijakesksus**

Õpetaja üks peamisi eesmärke on toetada ja julgustada ennastjuhtivat õppimist, võimaldades saada iseseisvaks, motiveeritud, oma õpingute eest vastutavaks õppijaks, kes on võimeline ise otsustama kuidas ja mida õppida (King, 1999).

**Õppimine kui …**

**Probleemõpe**

Õppemeetod ja -strateegia, kuidas organiseerida õppimisprotsessi nii, et õppijad oleksid aktiivselt kaasatud probleemi lahendamisse (Graaff & Kolmos, 2007).

**Mentorlus**

Mentorlus on kahe inimese koostöösuhe, kus kogenud töötaja (mentor) jagab vabatahtlikult ning üldjuhul tasuta oma teadmisi ja kogemusi vähemkogenud töötajaga (menteega). Selle asemel, et öelda õppijatele täpselt, mida teha, suunatakse õppijaid mõtlema ja kui nad jõuavad esimeste lahendusteni, siis neid juhendavad kogenud mentorid (Billet, 1994: 10).

Kogenud mentorilt oodatakse mudeldamist (modelleerimist), treenimist, „karkude“ ja muude meetodite samm haaval kasutamist, mida õppijad vajavad autentsete ülesannete lahendamisel. Tavaliselt alustatakse lihtsamatest ülesannetest ning lõpuks liigutakse oluliste tuumikülesannete (tuumikoskuste) poole.

# Kuidas seda kõike rakendada nooremtarkvaraarendaja õppeainetes?

JOBIT projekt soovitab kasutada konstruktivistlikku õpiteooriat (Wolf, 2005), mida kohandada meetoditega, mis kõige paremini täidavad kutsepedagoogika eesmärke. Konstruktivistlik käsitlus ühendab õppijakesksuse, mille eesmärk on konstrueerida konkreetse teemaga seotud teadmisi, aga on laiendatav katmaks teemat ulatuslikumalt.

Etapid:

**Algatus**

Vali hoolikalt õpiväljunditega seotud õppematerjalid (konspektid, videod, esitlused), mille abil on õppijatel võimalik interaktiivselt õppida. Õppematerjalidega tutvumine on õppija esimene iseseisev vaade teemale, esmamulje ja esmased arusaamad.

**Teemakäsitlus**

Teema edasine käsitlemine toimub klassiruumis õppijate ühisaruteluna või seminarina. Õpetajal võib olla moderaatori roll, ta võib sekkuda, täpsemalt selgitada või määratleda keerukamaid või huvitavamaid teemasid. Antud etapp on ettevalmistus praktiliseks tegevuseks stimuleerimaks ja tõhustamaks õppimist.

**Juhendatud praktiline töö**

Õppematerjalide käsitlemisele peaks ideaalsel juhul järgnema juhendatud praktiline töö. Õpetaja roll on olla mentoriks ja õppijaid oodatakse järgima ning läbi praktiliste tegevuste omandatakse õpiväljundid. Sõltumata sellest, kas edaspidi toimub töö individuaalselt või grupis, praktilised tegevused on eelduseks edasisele iseseisvale tööle.

**Individuaalne või rühmatöö**

Sellel etapil on õppijatel võimalus töötada iseseisvalt ja praktiseerida seda, mida eelnevalt õpiti. Ülesanded võivad olla individuaalsed (sissejuhatavad ja algkursused) või rühmatööd (edasijõudnud). Rühmatööde tegemisel tuleb arvestada rühmatöö tüüpiliste probleemidega, nende ennetamiseks tuleks püstitada selgepiirilised eesmärgid ja nõuda, et kõik teeksid ja vastutaksid oma osa eest.

**Õppimise hindamine**

Individuaalsete ülesannete täitmine (mitte kopeerimine) peab demonstreerib õpiväljundite piisava taseme saavutamist. Rühmatöö puhul tuleks hoolikalt planeerida, määrata spetsiifilised eesmärgid, püstitada igale õppijatele rühmas oma ülesanne ja vastutus, et kindlustada kõikide rühmaliikmete õppimine.

# Scrum

## Metoodika

Scrum on agiilne metoodika, mida kasutatakse keerulistes projektides. Algsel kasutati Scrumi tarkvaraarendusprojektides, kuid see sobib kasutamiseks mistahes valdkonna keerukate ja innovatiivse skoobiga tööde puhul. Scrumi kasutatakse laialdaselt IT valdkonnas, aga üha enam ja enam katsetatakse ka alternatiivseid valdkondi, näiteks hariduses (EduScrum Guidebook).

Scrum baseerub empiiriliste protsesside juhtimisteoorial, mille aluseks on põhimõtted, et teadmine sünnib kogemusest ja otsuseid saab teha vaid olemasoleva informatsiooni põhjal. Scrum on iteratiivse ja inkrementaalse lähenemisega, optimeerib etteennustatavust ja juhib riske. Juhtimine toetub kolmele sambale: läbipaistvus, ülevaatus ja paindlikkus.

**Läbipaistvus**: protsessi olulisemad aspektid peavad olema tulemuste eest vastutajatele nähtavad. Läbipaistvuse saavutamiseks peavad need aspektid olema standardselt defineeritud, et vaatlejad saaksid tulemusi hinnata ja jõuaksid tulemuste osas ühisele arvamusele.

**Ülevaatus (inspekteerimine, inspection)**: Scrumi kasutajad peavad perioodiliselt üle vaatama Scrumi tseremooniad ja monitoorima projekti edenemist, et avastada soovimatuid asjaolusid. Tulemuste üle vaatamine ei pea toimuma nii sageli, et see hakkaks tööd segama. Ülevaatusi tuleks teha töökeskkonnas asjatundlikult ja hoolikalt.

**Paindlikkus**: kui selgub, et protsessides on kõrvalekaldeid või vigu ning tulemus ei ole vastuvõetav, siis tuleb protsessi muuta. Muudatused tuleb teha võimalikult kiiresti, et minimeerida edasisi kõrvalekaldeid.

Scrum kirjeldab 4 formaalset sündmust, mille abil toimub inspekteerimine ja muudatuste sisse viimine:

1. sprindi planeerimine
2. igapäevane koosolek
3. sprindi üle vaatamine ja
4. sprindi retrospektiiv

## Scrumi rollid

Scrum määratleb 3 rolli:

* **Toote omanik,** kes vastutab **toote väärtuse maksimeerimise** ja meeskonna töö tulemuste eest. Enamasti võib õpetaja võtta toote omaniku rolli ja ta vastutab üldjuhul järgmiste tegevuste eest:
  + Eesmärkide seadmine, kasutuslugude (*user stories*) koostamine, probleemi defineerimine (algatus), tähenduse loomine (lühiülevaate etapp)
  + Ajaplaneerimine
  + Monitoorimine ja kvaliteedi kindlustamine
  + Hindamine ja tagasiside
  + Pigem treener või mentor kui juht, õpetaja ei õpeta, aga vajadusel aitab.
* **Scrum Master** vastutab, et meeskond järgib Scrumi põhimõtteid ja reegleid. Scrum Master on **meeskonda teenindav liider**, meeskonna ja väliste osapoolte vahendaja (va toote omaniku vahendaja), toetab tegevusi, mis meeskonda aitavad maksimeerides nii meeskonna poolt loodud väärtust. Scrum Masteri ülesanded on:
  + Inspireerida ja innustada meeskonda.
  + Uuendada Scrum boardi.
  + Algatada koosolekuid.
  + Võimaldada meeskonnal töötada ja saavutada maksimaalne tulemus.
* **Meeskond (arendajad)** peab olema iseorganiseeruv ja mitmekülgsete oksustega. Scrumi tiim peaks moodustuma nii, ta oleks optimaalselt paindlik, loov ja tulemuslik. Meeskond annab toodangut üle iteratiivselt (et saada maksimaalselt tagasisidet) ja inkrementaalselt (tagades, et tellijal oleks alati olemas kasulik ja töötav toote versioon). Meeskond:
  + Koosneb 4 kuni 9 liikmest, kelle osakused täiendavad üksteist.
  + Vastutab ise oma töökorralduse eest.
  + Liikmed on võrdsed (meeskonna töö on kõige olulisem).

Scrum määratluses on sprint ajaperiood (kestus sõltub asjaoludes), mille jooksul meeskond otsustab teha valmis projekti alamosa (*subset*).

## Scrum hariduses

Scrumi kasutati hariduses esmakordselt Hollandis (EduScrum). Scrumi kasutamine hariduses annab õppijatele enam võimalusi olla kaasatud oma õpingutesse. Kasutades rohkem praktilisi projektitöid saavad õppijad õppida efektiivsemalt ja ennast arendada, et vastata töömaailma ootustele. Scrum sobib erinevas vanuses ja erinevate oskustega õppijatele ning õpetab vastutust. Projektis JOBIT kasutame Scrumi kui tarkvaraarenduse mudelit, aga ka kui aktiivõppemeetodit. Õppimisprotsessis keskendume oskustele, mida õppijad tulevikus vajavad. Järgnevas tabelis on toodud olulisemad tulevikus tarvilikud oskused.

|  |  |  |  |
| --- | --- | --- | --- |
| Suhtlemisoskus ja -julgus | Ettevõtlikkus | Vastutustunne, otsustamine | Õppima õppimine |
| Iseseisvalt töötamine | Lugemisoskus, funktsionaalne lugemine | Eneseanalüüs | Rutiinitaluvus |
| Ajaplaneerimine | Eetilisus | Meeskonnatöö | Paindlikkus |
| Probleemide lahendamine | Keskendumisoskus | Informatsiooni otsimine | Enesemotivatsioon |

Kutseõppes on Scrum hästi kasutatav või mõnel juhul vähemalt integreeritav. Scrumi kasutamise positiivsed küljed õppijate jaoks:

1. ülesanded on praktilised;
2. õpetab ettevõtlikkust ja algatamist;
3. õpetab mõistma meeskonnatööd ja töötama meeskonnas;
4. õppimine on praktiline, soovitav kaasata eriala spetsialiste, näiteks teha töötube;
5. sobib erinevatele vanusegruppidele (meeskonnas on vähe liikmeid);
6. õpetab ülesandeid eristama (*user stories*) ja delegeerima.

Tarkvaraarenduse eriala õpilased peavad õppima kasutama Scrumi, sest see on üks enamkasutavaid tarkvara arendusmetoodikaid. Projektis JOBIT kasutatakse seda kui aktiivõppe ja praktika meetodit. Scrumi kasutamise kõige olulisem mõju on see, et kasvab õppijate vastutustundlikkus – õppijad tunnetavad, et nad ise juhivad ja vastutavad oma õppimise eest. Scrum ühendab õppijaid ja aitab õpetajal visuaalselt õppimisprotsessi jälgida. Scrum suurendab õppijate:

1. ettevõtlikkust ja proaktiivsust;
2. efektiivsust;
3. suhtlemisoskust;
4. loovust / innovatiivsust / koosloovust / enesekindlust;
5. projektijuhtimise oskust;
6. õppimine muutub õppija enda omaks.

## Kuidas kasutada Scrumi kutseõppes?

Järgnevalt on kirjeldatud Scrumi samm-sammulist kasutamist näiteks tarkvara arendusprojektis:

Samm 1: **meeskondade moodustamine** –

1. klass otsustab, kellest saab Scrum Master;
2. Scrum Master valib/moodustab meeskonna;
3. Igaüks kirjutab anonüümselt märkmepaberil oma olulised oskused (näiteks 3), Scrum Master moodustab nende põhjal meeskonna nii, et liikmete oskused täiendaksid üksteist.

Samm 2: **toote (tarkvara) planeerimine ja ajakava** (algatus + teemakäsitlus) –

1. eesmärkide seadmine,
2. kirjutada ja analüüsida ülesanded (kasutuslood ehk user storied),
3. koostada ajakava,
4. määrata hindamise põhimõtted ja viisid,
5. leida vajalikud ja soovituslikud õppematerjalid

Samm 3: **sprindi planeerimine** –

1. sprindi jaotamine ülesanneteks (user stories),
2. Scrum Boardi koostamine st ülesanded töölauale,
3. töölaual peab olema minimaalselt veerud: ToDo (teha), Busy (tegemisel), Done (tehtud),
4. töölaud võib olla elektrooniline (e-board).

Samm 4: **koosolekud (*stand-up meetings*)** –

1. kestus 2-4 minutit;
2. kõikide osalemine on oluline;
3. mis oli viimane asi, mida meeskond tegi (või mida Sina meeskonnas tegid)?;
4. mida meeskond teeb järgmisena?
5. millised on probleemid? Kuidas neid lahendada?

Samm 5: **sprindi tulemus (*release*) ülevaatamine**  (õppimise hindamine) –

1. kiire ülevaade, näiteks poster-esitlus;
2. kuulajate/vaatajate tagasiside;
3. õppida teistelt, jagada ja õpetada/abistada meeskonnaliikmeid.

Samm 6: **tagasivaade** **(retrospective)** **sprindile** –

1. tagasi sammule 3 (sprindi planeerimine);
2. analüüsida tulemusi, hinnata meeskonna tööd; hinnata ja väärtustada suhteid; hinnata protsessi;
3. mis läks hästi, mis läks halvasti, potentsiaalsed parandamiskohad.

# Kasutatud allikad

Kasutatud Scrumi koolituse materjale, veebruar 2017 Kehtnas, koolitajad Harri Hautala ja Linda Salminen, Helsinki Business College.

eduScrum homepage <http://eduscrum.nl/en/>

The eduScrum Guide <http://eduscrum.nl/file/CKFiles/The_eduScrum_Guide_EN_December_2013_1.0.pdf>

Linders, B. (2013). Scrum for Education - Experiences from eduScrum and Blueprint Education. [Electronic Version]. Available from: <https://www.infoq.com/articles/scrum-education>

Sutherland, J. (2012). Scrum: The Future for Education? [Electronic Version]. Available from: <https://www.scruminc.com/scrum-future-for-education-2/>

Solingen, R. A view into the future – using Scrum as teaching tool in high school (and prior?) [Electronic Version]. Available from: <http://rinivansolingen.nl/a-view-into-the-future-using-scrum-as-teaching-tool-in-high-school-and-prior/?lang=en>

The Scrum Guide - <https://www.scrumalliance.org/why-scrum/scrum-guide>

The Beginner’s Guide to Scrum and Agile Project Management <http://blog.trello.com/beginners-guide-scrum-and-agile-project-management>

# LISA 1 – Tunnikavade näidised

|  |  |
| --- | --- |
|  | Tunnikava - Programmeerimine |
|  |  |
| Teema | Objektorienteeritud programmeerimise põhimõisted, klassi loomine ja kasutamine |
| Eeldused | Arvutid, milles on olemas arenduskeskkond (IDE) |
| Õppijate arv | 10-24 (töötavad individuaalselt või paaris) |
| Kestus | * 150 minutit klassiruumis sh paus 15-20. * Iseseisev töö (enne põhiteemat) 20-30 minutit õppematerjalidega tutvumiseks. * Iseseisev töö (pärast põhiteemat) 1-3 tundi. |
| Näidised | Class Persoon, class Osakond, class Tudeng |
| Töölehed | [Valikuline] Täida klassi faili väljad klassidega seotud põhisüntaksi omandamiseks |
| Näidisvideo | <https://www.youtube.com/watch?v=SS-9y0H3Si8> |
| Korraldus | Õppijad töötavad õpetaja juhendamisel, võivad moodustada paare tegevuste assimileerimiseks. |
| Teadmised ja oskused | * Alusteadmised keele süntaksist * Andmetüübid * Protseduurne programmeerimine (meetodid/funktsioonid/alamprogrammid) |
|  |  |
| Skoop | Tutvustada õppijatele objektorienteeritud lähenemist, objekti mõistet ja õpetada neid looma oma klasse. |
| Õpiväljundid | 1. Teab, et objekt on mõiste, mis tähistab individuaalset olemit. 2. Mõistab objekti omaduste (atribuutide) tähendust. 3. Mõistab objektide käitumist (*methods*). 4. Mõistab ligipääsumeetodite kasutamist ( [[1]](#footnote-1) ) 5. Luua klassi fail.    1. Redigeerida ja luua tarvilikke faile.    2. Anda klassi atribuutidele väärtused.    3. Loob ligipääsumeetodid (*access methods* ( 1 )    4. Loob constructor ja/või destructor meetodid ( 1 )    5. Loob teisi meetodeid. 6. Loob objekte, oskab muuta nende omadusi, loob ja kasutab klassi meetodeid. |
|  |  |
|  | Eelnevad tegevused |
| Algatus | Õppematerjal peab olema õppijale kättesaadav enne klassiloenguid. See võib toimuda klassiruumis või e-õppekeskkonnas. Peamine eesmärk on tutvustada objektorienteeritud programmeerimise olemust ja põhimõisteid. Näidisõppematerjalid:   * Lühike esitlus, mis demonstreerib klasside vajadust, soovitavalt siduda spetsiaalsete andmetüüpidega, mida kasutatakse reaalsetes ärilahendustes. * Õppevideo (3 -5 minutit) selgitamaks, mis on objektorienteeritud programmeerimine, kuidas kogu reaalne maailm koosneb objektidest koos oma omaduste ja meetoditega. * [Valikuline] lingid muudele õppevideotele. * [Valikuline] soovituslikud õppematerjalid lugemiseks. * [Valikuline] lingid e-õppematerjalidele. |
|  |  |
|  | Põhiteema (klassiruumis) |
|  | Õpetaja peab olema veendunud, et kõik õppijad on õppematerjalidega tutvunud.  [Valikuline] Kui juhtumisi kõik õppijad pole ettevalmistunud, siis peab õpetaja tegema 5-10 minutilise ülevaate selgitamaks objektorienteeritud programmeerimise olemust ja põhimõisteid. |
| Teemakäsitlus | Õpetaja alustab vestlust, milles õppijad on kutsutud arutlema teemal objektorienteeritus ja ise sõnastama oma arusaamist teemast.  Tegevus: õppijad peaksid tooma enda näiteid:   * klassidest, * klassi omadustest, * klassi meetoditest, * klasside reaalsest kasutamisest.   Tegevus: õpetaja võib tuua enda näiteid, parandada vigu, aga õppijad võivad ka ise (näiteks paaris töötades) otsustada, kas toodud näide on õige või vale. |
|  | Õpetaja peab olema veendunud, et kõik õppijad on teemast aru saanud ja on valmis õppima klasside praktilist loomist. |
| Juhendatud praktiline töö | Esmalt õpetaja demonstreerib lühikeste näidete abil, kuidas klass defineeritakse (näiteks klass <Persoon> omadustega <nimi> ja <telefon>). Algul peaksid õppijad õpetajat jälgima, mitte püüda iseseisvalt tegutseda. Samuti võiks olla ettevalmistud üks või kaks näidisprogrammi, mis demonstreeriks klassi kasutamist ja veel ühe või mitme objekti loomist.  Tegevus: õpetaja koos õppijatega teevad veel ühe näite (näiteks, klass <Osakond>).  Tegevus: õppijad moodustavad paarid ja realiseerivad kolmanda klassi (näiteks klass <Tudeng> ). Õpetaja abistab vajaduse korral.  Tegevus: õppijad töötavad individuaalselt, loovad neljanda klassi. Näidisklass võiks olla tehtud õppijate endi ettepanekuna, kasutada võib lühiülevaates toodud näiteid, aga see klass peaks kindlasti olema selline, mida õppijad hästi tunnevad (näiteks mobiil, tahvel, arvuti vms) |
|  | Õpetaja veendub, et kõik õppijad on väljundid omandanud, küsib kas on tarvis midagi täpsustada, kas mõni teema jäi arusaamatuks. |
|  | Lõpetuseks teeb õpetaja lühiülevaate läbitud teemadest. |
|  | Põhiteema järel |
| Individuaalne või rühmatöö | Õppijad töötavad individuaalselt. Õpetaja annab piisava mahuga iseseisva töö ülesanded klasside kasutamise harjutamiseks.  [Valikuline] iseseisva töö ülesanne võiks sisaldada mitut väiksemat klassi, mis annab võimaluse harjutada klasside loomist.  [Valikuline] iseseisva töö ülesanne võiks olla ühe suure reaalse maailma klassi loomine, mis võimaldaks õppijatel katsetada klassi kavandamist, loomist ja kasutamist.  Ülesanne peaks sisaldama ka programmi koostamist, mis loodud klasse kasutab. Programmi funktsionaalsus peaks olema autentne, idee/probleem võiks pärineda reaalsest elust, et õppija saaks leida probleemile oma lahenduse.  [Valikuline] selleks, et vältida või minimeerida lahenduste kopeerimist, peaks õpetaja pakkuma kaks või enam kodutööde varianti, variandid määratakse kasutades kindlaid valikukriteeriume. Näiteks kui on olemas kaks sama raskusastmega varianti, siis kasutades õppijate isikukoode, kui isikukoodi viimane number on paaris, siis tehakse variant A ja paaritu arvu korral tuleb teha variant B.  Iseseisva töö ülesanded peaksid selle teema puhul olema individuaalsed. Väga oluline on, et iga õppija õpiks ja saavutaks õpiväljundid ise, et edaspidi luua klasse iseseisvalt ja lähtuda objektorienteeritud programmeerimise põhimõtetest.  Juhtnöörid/nõuanded:   * Alati tuleks eelnevalt selgitada iseseisvate tööde hindamiskriteeriume. * Anna piisavalt aega iseseisvateks töödeks, arvestades õppijate koormust ja muid faktoreid, mis võiksid kursuse ajakava mõjutada (näiteks pühad jms) . * Tuleb arvestada ka sellega, et õppijad kopeerivad lahendusi muudest allikatest. Õpetaja peaks meenutama, et õppijate endi huvides mitte kopeerida, sest siis nad ju tegelikult ei õpi ja eksamil või tööl ei saa nad lahendamisega hakkama. Õpetaja peaks ka selgitama, et selline käitumine on ebaeetiline ja vastuvõtmatu. * Kui tähtaeg on ületatud, siis teeb õpetaja kättesaadavaks oma lahenduse. See võimaldab õppijatel aru saada, mida neilt oodati. * Tööde parandamisel tuleb leitud vead kommenteerida. Kommentaarid ja hinnangud peavad olema õiglased, põhinema hindamiskriteeriumitel, kõik hinnangud peavad olema põhjendatud. Õppijad peaksid kommentaaride abil mõistma, mis läks valesti ja mida tuleks edaspidi teha teisiti. |
| Tulemuste hindamine | Kui õppijad on lõpetanud kõik ülesanded, siis peaksid nad olema saavutanud õpiväljundite rahuldava taseme.  [Valikuline] iseseisvate tööde esitamisele vahetult järgnevas tunnis võiks teha veebipõhise testi, mis käsitleb läbitud teemat.  [Valikuline] vahetult järgnevas tunnis võib teha lühikese testi, et hinnata õpiväljundite saavutamise taset. Selleks võib olla [valikuliselt]   * klassiruumis täidetav tööleht, * klassi loomise ja kasutamise lühikirjeldus, * programmi koostamine etteantud klassi kasutamiseks, * kirjeldada etteantud klassi baasil kirjutatud programmi väljund. |

|  |  |
| --- | --- |
|  | Tunnikava näidis – Tarkvaraarendus |
|  |  |
| Teema | Tarkvara nõuete koostamine |
| Eeldused | Paber ja pliiats |
| Õppijate arv | 10-24 (individuaalne, paaris- või meeskonnatöö) |
| Kestus | * 150 minutit klassiruumis sh paus 15-20 minutit. * Iseseisev töö (enne põhiteemat) 20-30 minutit (õppematerjalidega tutvumine) * Iseseisev töö (pärast põhiteemat) 2-4 tundi. |
| Näidisdokumendid | puudub |
| Töölehed | [Valikuline] |
| Näidisvideod | <https://www.youtube.com/watch?v=vSXn16qMEZo>  <https://www.youtube.com/watch?v=XUDtUpzxvvw> |
| Korraldus | Sõltub olukorrast |
| Teadmised ja oskused | * Nõuete koostamine ja esialgne analüüs tarkvaraarendusprojekti käivitamiseks. |
|  |  |
| Skoop | Tutvuda nõuete välja selgitamise tehnikatega ja erinevate lähenemistega. |
| Õpiväljundid | 1. Teab, mis on otsesed nõuded. 2. Teab, mis on kaudsed (varjatud) nõuded. 3. Teab millised on olulisemad kvalitatiivsed tehnikad nõuete välja selgitamiseks. 4. Teab millised on kvantitatiivsed tehnikad nõuete välja selgitamiseks. 5. Viib läbi ajurünnakut. 6. [Valikuline] Viib läbi intervjuu nõuete välja selgitamiseks 7. Teostab dokumendivaatlust ja –analüüsi. |
|  |  |
|  | Eelnevad tegevused |
| Algatus | Õppijatel peaksid olema põhiteadmised ja arusaam, kuidas ja mil viisil selgitakse välja nõuded tarkvarale, mille poolest eristuvad otsesed ja kaudsed nõuded. Õppematerjalid peavad olema õppijatele kättesaadavad enne põhiteemat e-õppekeskkonnas. Soovituslikud õppematerjalid:   * Lühike esitlus tutvustamaks nõuete välja selgitamist ja selleks kasutavaid metoodikaid. * Video (piisavalt lühike) selgitamaks kaasnevaid teemasid. * [Valikuline] lingid soovituslikele videotele. * [Valikuline] lingid e-õppematerjalidele.   [Variant A] Õpetaja võiks kasutada oma töömaailma kontakte, et saada mõne reaalse tarkvaraprojekti/probleemi kirjeldus.  [Variant B] Õpetaja võiks kasutada oma töömaailma kontakte, et kutsuda tundi potentsiaalne tellija, kes selgitaks oma tarkvara vajadust või kirjeldaks reaalset probleemi.  Mõlema variandi puhul peavad õppijad saama kasutada kõiki vajalikke materjale, et saada paremini aru tellija vajadustest. |
|  |  |
|  | Põhiteema (klassiruumis) |
|  | Õpetaja peab veenduma, et õppijad on õppematerjalidega tutvunud.  [Valikuline] Kui juhtumisi kõik õppijad pole ettevalmistunud, siis peab õpetaja tegema 5-10 minutilise ülevaate selgitamaks objektorienteeritud programmeerimise olemust ja põhimõisteid. |
| Teemakäsitlus | Õpetaja alustab vestlust, milles õppijad on kutsutud arutlema ja sõnastama oma arusaamist teemal tarkvara nõuete koostamine.  Arutelu käigus peaks ilmnema suur hulk erinevaid tehnikaid ja õppijail palutakse rühmitada tehnikad vastavalt sellele, mis tüüpi nõuete puhul, mis tüüpi tehnikaid kasutada. |
|  | Õpetaja peab veenduma, et kõikidel õppijatel on teema selge ja võib alustada praktilise tööga. |
| Juhendatud praktiline töö | [Vaikimisi] Õpetaja võtab endale tellija rolli. …  [Variant A] protsessis osaleb tarkvaraarendusfirma esindaja, …  … kes soovib arendada või luua teatud tarkvaratoodet. Õppijad proovivad kasutada erinevaid tehnikaid nõuete väljaselgitamiseks, oluline on erinevate tehnikate proovimine, mitte niivõrd absoluutselt kõikehõlmava nõuetekomplekti koostamine.  Tegevus: õppijad ja firma esindaja osalevad ajurünnakus, et määratleda projekti eesmärgid, tellija vajadused ja koostada nõuded tarkvarale  Tegevus: Firma esindaja esitleb tüüpdokumente, mida kasutatakse reaalses arenduses ja selgitab nende kasutamist. Edaspidi tuleb teha dokumendi analüüs, siis kui esindaja on lahkunud või iseseisva tööna.  [Valikuline] Tegevus: õppijad teevad intervjuu firma esindajaga, et välja selgitada varjatud nõudeid. Intervjuu võib toimuda õpetaja juhendamisel, küsimused koostavad õppijad ise. Samas tuleb märkida, et selles faasis ei ole õppijad veel valmis ise tellijat intervjueerima.  Tegevus: õpetaja võib anda intervjuuks küsimused ette ja õppijad (individuaalselt või rühmatööna) otsustavad, millised küsimused antud olukorras sobivad.  Tegevus: juhul kui kasutatakse varianti B, siis võib õppijad jaotada rühmadeks, et luua fookusgrupid ja selgitada välja nende nõudmised.  Tegevus: juhul kui kasutatakse varianti B, siis võiks olla spetsiaalselt ettevalmistatud lühike koosolek, kus tulevad kokku firma esindaja ja õppijad ning nõuded formuleeritakse koos. |
|  | Lõpetuseks teeb õpetaja kokkuvõtte saadud teoreetilistest teadmistest ja praktilistest tegevustest. |
|  | Peateema järgselt |
| Individuaalne või rühmatöö | Õppijad võivad töötada individuaalselt või eelnevalt moodustatud rühmadena, et lahendada järgmised ülesanded:   * Vastavalt õpetaja kirjeldatud või firma esindaja pakutud projektile:   + Analüüsida olemasolevaid dokumente ja määratleda dokumentides kirjeldatud nõuded.   + Koostada nõuete dokumendi mustand, milles on üldjoontes kirjeldatud tarkvara nõuded. * Kõikide variantide puhul:   + Vii läbi uuring ja koosta intervjuu küsimused tellija intervjueerimiseks.   + Vii läbi uuring ja koosta küsimustik, mida saaks kasutada huvigruppide ja tarkvara kasutajate küsitlemiseks.   Juhtnöörid/soovitused:  Tööde parandamisel tuleb leitud vead/puudujäägid kommenteerida. Kommentaarid ja hinnangud peavad olema õiglased, põhinema hindamiskriteeriumitel, kõik hinnangud peavad olema põhjendatud. Õppijad saavad kommentaare kasutada mõistmaks, mis läks valesti ja mida tuleks edaspidi teha teisiti.   * Alati tuleks eelnevalt selgitada iseseisvate tööde hindamiskriteeriume. * Anna piisavalt aega iseseisvateks töödeks, arvestades õppijate koormust ja muid faktoreid, mis võiksid kursuse ajakava mõjutada (näiteks pühad jms) . * Tuleb arvestada ka sellega, et õppijad kopeerivad lahendusi muudest allikatest. Õpetaja peaks meenutama, et õppijate endi huvides mitte kopeerida, sest siis nad ju tegelikult ei õpi ja eksamil või tööl ei saa nad lahendamisega hakkama. Õpetaja peaks ka selgitama, et selline käitumine on ebaeetiline ja vastuvõtmatu. |
| Tulemuste hindamine | Enne järgmist klassiloengut võiksid õppijad teha veebipõhise testi, et hinnata õpiväljundite saavutamise taset. Test võib olla enesekontrollitest.  [Valikuline] Õpetaja peaks kõikidele õppijatele andma iseseisvatele tööde kohta tagasisidet. Järgmises tunnis peaks õpetaja koos õppijatega tagasiside kokkuvõtet arutama. |

1. Valikuline – sõltub kasutatavast programmeerimiskeelest [↑](#footnote-ref-1)