Sissejuhatus tarkvaraarendusse

Martti Raavel

martti.raavel@tlu.ee

Esimene loeng

- Sissejuhatus
- Tarkvara
- Tarkvaraarendus
- Tarkvaraarenduse elutsükkel
- Git
- Github
- .gitignore
- Mardown
- Github issue

Sissejuhatus

- Aine ülesehitus
- Kodused tööd
- Hindamine

Aine ülesehitus

- Hindeline
- Loengud kohapeal
- 4 loengut + eksam
- Materjalid Github-is
- Salvestused
- Kodused tööd

Kodused tööd

- Kohustuslikud
- Tähtajaks!
- Kuni 50% lõpptulemusest
- Vigade leidmine materjalidest annab lisaboonust
- Materjalide täiendamine alnnab lisaboonust

Hindamine

- Eksam
  - 50% punktidest kodused tööd
  - 50% punktidest kohapeal tehtav avatud ja valikvastustega test

Mis on tarkvara?

Tarkvara on juhiste või programmide kogum, mis on loodud arvutisüsteemis konkreetsete ülesannete või funktsioonide täitmiseks. See on programmide, andmete ja juhiste kogum, mis ütleb arvutile, mida ja kuidas teha.

Tarkvara liigid

- operatsioonisüsteemid;
- rakendustarkvara;
- programmeerimiskeeled;
- utiliiditarkvara;
- jne.

Tarkvara eesmärk

- andmete haldamine;
- dokumentide loomine;
- graafika kujundamine;
- mängude mängimine;
- jne.

Avatud vs suletud lähtekoodiga tarkvara

- Mis neil vahet on?
- Kumb lähenemine on parem?

Kust tarkvara saab?

- Osta
- Ise teha
- Lasta teha

Plussid/miinused?

Mis on tarkvaraarendus?

Tarkvaraarendus on tarkvararakenduste kavandamise, loomise, testimise ja hooldamise protsess. See hõlmab programmeerimiskeelte, tarkvaraarendustööriistade ja parimate tavade kasutamist, et luua tarkvara, mis vastab konkreetsetele nõuetele ja lahendab konkreetseid probleeme.

Tarkvaraarenduse elutsükkel

- Planeerimine
- Nõuete määramine
- Disain
- Programmeerimine
- Testimine
- Evitamine ja hooldus

Planeerimine

- Mida?
- Miks?
- Kelle jaoks?

Nõuete määramine

- Täpsemalt mida vaja
- Prioritiseerimine
- Soov ei võrdu vajadus
- Kliendi kinnitus

Disain

- Arhitektuur
- UX
- UI

Programmeerimine

- Mis operatsioonisüsteemile?
- Mis keeles?

Testimine

- Manuaalne testimine
- Automaattestimine

Töösse andmine

- CI/CD
- Koolitus
- Dokumentatsioon

Hooldus

- Monitoorimine
- Vigade parandus

Kuidas kirjutatud koodi kaitsta ja hallata?

- Palu faile
- Palju muudatusi
- Rohkem kui üks arendaja/osapool

Versioonikontroll

Versioonikontroll on süsteem, mis jälgib ja haldab aja jooksul failides tehtud muudatusi.

Miks versioonikontroll?

- Koostöö
- Ajalugu ja jälgimine
- Muudatuste tagasivõtmine
- Hargnemine ja ühendamine

Git

Git on hajutatud versioonihaldussüsteem, mis on loodud tarkvara arendamise käigus lähtekoodi muutuste jälgimiseks.

Git-i sõnavara

- Repository
- Clone
- Pull
- Branch
- Commit
- Push
- Pull request
- Merge

Git-i töövoog

Giti töövoog on parimate tavade ja juhiste kogum Giti kasutamiseks koodimuudatuste haldamisel. Giti töövooge on palju, kuid kõige levinumat neist nimetatakse *feature branch flow*,

Feature branch flow

- Loo uus haru (*branch*)
- Kirjuta koodi
- Testi kood
- *Commit*i muudatused
- Push
- Pull request
- Merge
- Kustuta haru

Harud

- main
- dev
- test
- alamharud

Github

GitHub on veebipõhine platvorm, mida kasutatakse versioonikontrolliks ja koostööks tarkvara arendamiseks. Github pakub Giti versioonikontrollisüsteemi kasutavate tarkvaraarendusprojektide hostimisteenust.

Gtihubi funktsioonid

- Hoidla majutus
- Koostöö tööriistad
- Juurdepääsukontroll
- Integratsioonid
- Sotsiaalsed funktsioonid

Githubi kasutamine

- git CLI (*Command Line Interface*)
- Graafilise kasutajaliidesega tööriist
  - Github Desktop

Githubi kasutamise harjutamine 1

- Loo omale repositoorium
- Klooni see oma arvutisse
- Lisa sinna README.md
- Täida loodud fail mingi sisuga
- Tee *commit*
- Push

Githubi kasutamise harjutamine 2

- Tee uus haru
- Lisa mingi uus fail
- Tee commit
- Tee push
- Tee Pull request
- Tee merge

Koduste tööde reposse oma kausta tegemine

- Clone
- Tee kaust Eesnimi_Perekonnanimi
- Sinna alla loo fail README.md, sinna lisa oma nimi
- Loo kaust Programmeerimine_I
- Sinna alla loo fail README.md
- Loo kaust SJTA
- Sinna alla loo fail README.md
- Commit
- Push
- Pull request

.gitignore

`.gitignore` on konfiguratsioonifail, mida Git kasutab, et määrata, millised failid ja kataloogid tuleks versioonihaldussüsteemist välja jätta.

Markdown

Markdown on märgendikeel, mis võimaldab kasutajatel kirjutada lihtteksti ja vormindada seda lihtsa süntaksiga, et luua dokumente, mida on lihtne lugeda ja kirjutada.

Githbu Issue

GitHubi kontekstis on *issue* omadus, mis võimaldab kasutajatel jälgida konkreetse hoidla ülesandeid, vigu ja funktsioonitaotlusi

Kodune töö

- Endanimelises kaustas README.md lühikese ülevaade oma kodutööde kohta
  - Kuidas ülesande lahendamisele lähenesid
  - Kas oli probleeme
  - Kuidas probleemid lahendasid
- Programmeerimine I ülesanded
- Oma Programmeerimine I kausta
  - Kodune_1
  - Kodune_2