Eksamiküsimuste vastused aines Tarkvaratehnika (MTAT.03.094)
Aeg: 11. jaanuar 10:00 - 14:00

Küsimuste & vastuste redaktor: Oleg Mürk (oleg.myrk@gmail.com)

Põhimõtted:
- Lühiküsimuste eesmärk on teadmiste kontroll. Vastused küsimustele leiab loengumaterjalidest.
- Lühiküsimuste hindamisel on ainus kriteerium vastaja demonstreeritud teadmised.
- Seega nt küsimusele "mis on muster Observer?" antud vastus "see on nagu Swingi listenerid" on täiesti piisav.
- Pikkade küsimuste eesmärk on hinnata vastaja oskused analüüsis, projekteerimises, ja testimises, mida sai omandada lahendades praktikumi ülesandeid.

1) (3 p.)
Küsimus:
Mis on metoodika tarkvaraarenduses? Nimeta 3 tuntud metoodikat. Millised on nende plussid ja miinused?

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_04.pdf
Slaidid 14-18
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_07.pdf

Vastus:
Metoodika on süstemaatiline viis tarkvara arendamiseks. Näiteks XP, RUP, Waterfall. XP & RUP on iteratiivsed, Waterfall mitte. XP on väle (vähe plaane & dokumentatsiooni, adaptiivne), Waterfall plaan-juhitud, RUPi saab konfigureerida nii väledaks kui ka plaan-juhituks. Väledad metoodikad sobivad paremeni kui muudatuste hind on väike, nõuded muutuvad tihti, produktiivsus on tähtsam kvaliteedist. Plaan-juhitud metoodikad sobivad paremeni olukorda, kus muudatuse hind on kallis, nõuded muutuvad vähem, kvaliteet on tähtsam produktiivsusest.

2) (3 p.)
Küsimus:
Kirjelda lühidalt klassiskeemi olemust ning kirjelda mis tingimustel selle kasutamine on põhjendatud. Koosta lühike näide.

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_08.pdf
Slaid 8
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/uml.zuml

3) (3 p.)
Küsimus:
Kirjelda vähemalt 3 halva disaini tunnust (ik *design smells*). Too lühikesed näited.

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_15.pdf
Slaidid 7, 8

4) (3 p.)
Küsimus:
Selgita disaini mustrite (ik *design pattern*) olemust. Miks on disaini mustrid kasulikud? Kirjelda lühidalt vähemalt 3 mustrit.

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_15.pdf
Slaidid 14-17

5) (3 p.)
Küsimus:
Selgita must- ning valge- kasttestimise olemust. Millised on nende erinevused?

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_18_intro.pdf
Slaid 22

6) (5 p.)
Küsimus:
Kirjelda versioonihaldussüsteemi (ik *version control system*) olemust. Milleks on see kasulik? Selgita mõiste haru (ik *branch*). Nimeta vähemalt kaks versioonihaldussüsteemi. Millised on nende erinevused?

Materjal:
http://ats.cs.ut.ee/courses/2005/tvt/pmwiki/uploads/Main/software_engineering_20.pdf
Slaidid 4-10

7) (10 p.)
Autor:
Ivo Mägi (ivo@webmedia.ee)

Ülesanne:

Kasutusloo koostamine.

Lahendus & hindamine:
Esimeses järjekorras käsitleda loovust ning põhjalikkust.

Selgelt ja loetavalt peavad olema välja toodud kõrvalstsenaariumid. Põhistsenaarium ei tohi nende sisse ära kaduda. Üleminekud põhi- ja kõrvalstsenaariumitest peavad olema selgelt arusaadavad. Näha peavad olema erinevad lõppolekud (makse edukalt sooritatud, makse ebaõnnestunud)

Ära peavad olema toodud tegutsejad (kasutaja, internetipank, väline internetipank, Eesti Pank, rahapesu talitluse register). Kasutada võib sünonüüme. Väliste partnerite osalemise korral peavad olema käsitletud tõrkesituatsioonid.

Konverteerimis- ja ümardamissituatsiooni korral peab olema selgelt välja toodud jäägiga käitumine – mitte niivõrd raamatupidamisreegleid järgides aga käsitledes erinevaid eriolukordi.

Koguhinne peaks moodustuma 50% lahenduse vastavusest kasutajate vajadusele (kaetud on kõik harud, lahendus on hästi loetav ning ühtselt struktureeritud) ning 50% ülal eraldi välja toodud nõuete täitmisest.

8) (10 p.)
Autorid:
Erik Jõgi (erik.jogi@hansa.ee),
Oleg Mürk (oleg.myrk@gmail.com)

Ülesanne:
Koosta lihtsa tekstiredaktori disain.

Lahendus:
Põhimõtted:

- Kasutada tuleb *MVC* (*Model View Controller*) mustrit või vähemalt *DVA* (*Document View Architecture*), kus *View* ja *Controller* on kokku pantud.

- Tekstidokumendi *Model* peab olema selgelt eraldatud muust koodist. Lahendus peab kirjeldama:
  - kuidas esitatakse teksti,
  - kuidas määratakse positsioon tekstis

  ning põhioperatsioonid:
  - teksti suuruse küsimine (ridade arv, ridade pikkused),
  - rea sisu küsimine,

    - tähe sisestamine etteantud positsioonil,
    - reavahetuse sisestamine etteantud positsioonil,
    - kustutamine etteantud positsioonil või kahe positsiooni vahel,
    - teksti lõigu küsimine kahe positsiooni vahel,
    - teksti lõigu mestimine etteantud positsioonil,
    - laadimine & salvestamine failist.

- *Controlleris* peab olema selgelt välja toodud seisund:
    - kursori asukoht,
    - teksti sisestamine,
    - teksti valimine

  ning üldjoontes kuidas eri seisundites toimivad operatsioonid:
    - kursori liigutamine,
    - tähe sisestamine,
    - reavahetus sisestamine,
    - valimine,
    - kustutamine (tähe või valitud lõigu),
    - copy & paste.

- View kohta peab kirjeldama:
    - *viewporti* määramine (alates millisest reast ja veerust kuvada),
    - teksti kuvamise algoritm üldjoontes:
        - akna suuruse käsitlus,
        - tekstirea kõrguse käsitlus,
        - pikkade ridade käsitlus,
        - selekteeritud ala kuvamine
    - kursori paigutamise algoritm.

- Interatksioonid *Model*, *View*, *Controlleri* omavahel ja ülejäänud rakendusega:
    - kuidas View suhtleb Swingi akna komponendiga,
    - kuidas Controller saab teada klahvivajutustest,
    - kuidas View saab teada, et Model muutus ning tuleb ümber joonistada,
    - kuidas Controller määrab ja muudab View viewporti.

- Rakenduse ülesehitus:
    - mis juhtub rakenduse käivitamisel,
    - millised Swingi komponendid ja millal luuakse,
    - kuidas konfigureeritakse omavahel *Model*, *View*, ja *Controller* uue faili avamisel.

Hindamine:

- Vastus peaks sisaldama

    - klassiskeem või muu viis rakenduse struktuuri kirjeldamiseks,
    - käitumist kirjeldavad skeemid või struktureeritud tekst,
    - selgitav tekst,
    - olulisemate meetodite pseudokood.
- Püstituses mainitud mustrite nõue oli mõeldud vastaja tähelepanu pööramiseks nende kasutamise võimalusele. Tõenäolised mustrid on:
    - *Model View Controller* (*MVC*),
    - *Observer*,
    - *Factory*.
- Hinne peaks moodustuma proportsionaalselt ülaltoodud faktorite kaetusele ning esituse selgusele.
- Lisaks tuleb arvestada hea disani põhimõtetega mis olid kirjeldatud loengus ning ülesande püstituses.
- Swing API ning UML süntaksi detailide tundmine ei tohiks mõjutada hinnet.

9) (10 p.)
Autor:
Härmel Nestra (harmel.nestra@ut.ee)

Ülesanne:
Projekteerida testide komplekt failisüsteemilehitseja testimiseks musta kasti meetodil.

Lahendus:
Järgnevalt on kirjeldatud testide jaotused ekvivalentsiklassideks paljude tunnuste alusel. Konkreetne klass võib omakorda jaotuda klassideks. Sama tunnuse järgi võib klassideks mitut moodi jaotada.

Testide komplekt tuleks soovitavalt teha selline, et kui iga klassijaotuse järgi valida suvaliselt üks klass, nii et valitud klasside ühisosa on mittetühi, siis leidub komplektis test, mis kuulub sinna ühisossa. Niisugune testide komplekt tuleks liiga suur, et teda siia kirja panna, seepärast rahuldume klassijaotuste kirjeldamisega.

Kirjeldamisel on kasutatud minimalistlikku stiili. Tunnussõna järel on koolon ning selle järel on loetletud võimalik klassijaotus selle tunnuse alusel. Näiteks tunnus "Töö" jaotab võimalike testide ruumi klassideks kogu tehtava töö (st ühe testi) omaduste järgi. "Segatöö" on üks neist omadustest, st tunnus "Segatöö" jaotab klassideks ühe klassi tunnuse "Töö" järgi. Tunnus "Nimi" jaotab testid klassideks ühe faili või kataloogi nime omaduste järgi jne.

Töö:
- ainult selekteerimist kasutav,
- ainult raami kasutav,

- segatöö (mõlemat kasutav).

Töö:
- ühes eksemplaris,
- paljudes eksemplarides.

Segatöö:
- algul raam, siis selekteerimine,
- algul selekteerimine, siis raam,
- vaheldumisi.

Selekteerimist kasutava ja raami kasutava töö testide jaotus on üldjoontes sama. Siin esitame jaotuse korraga mõlema jaoks, pidades tunnuste ja klasside nimetuste valikul silmas eelkõige raami kirjutamist. Raami kirjutatud nimele vastab selekteerimisel valikute järjend. Lisand "(R)" tähendab, et antud jaotusklass eksisteerib vaid raami kirjutamise puhul, lisand "(S)" tähendab, et antud jaotusklass eksisteerib vaid selekteerimisel.

Nimi:
- kataloogideta nimi,
- üht kataloogi sisaldav nimi,
- paljusid katalooge sisaldav nimi.

Nimi:
- fail,
- kataloog.

Nimi:
- tühi (0 sümbolit),
- lühike (1 sümbol),
- pikim lubatud,
- lühim keelatud.

Nimi:
- tavasümbolitest koosnev,
- erisümboleid (tühik jne) sisaldav.

Katalooge sisaldav nimi:
- otseteega juurkataloogist eemale,
- otseteega juurkataloogi suunas,
- kõverteega algul juurkataloogi suunas, siis eemale,
- keerulisema teega.

Erisümboleid sisaldav:
- erisümboliga algav,
- erisümbolit keskel sisaldav,
- erisümboliga lõppev.

Fail:
- tühi,
- väike mittetühi,
- väga suur.

Fail:
- hetkel mitte kasutuses,
- parajasti teise programmi poolt loetav,
- parajasti teise programmi poolt kirjutatav.

Kataloog:
- tühi,
- sisaldab ühe faili,
- sisaldab parajasti niipalju faile, kui aknasse mahub,
- sisaldab pisut rohkem, kui aknasse mahub,
- sisaldab väga palju faile.

Fail või kataloog:
- eksisteerib,
- puudub eksisteerivas kataloogis,
- otsimistee vahekataloog puudub (R).

Fail või kataloog:
- on kuvatute seas esimene,
- on kuvatute seas viimane,
- valitakse pärast lühikest kerimist (S),
- valitakse pärast pikka kerimist (S),
- valitakse pärast edasi-tagasi kerimist (S).

Eksisteeriv fail või kataloog:
- avamisõigusega,
- avamisõiguseta, kuid avatava kataloogiga,
- avamisõiguseta otsimistee vahekataloogiga (R).

Eksisteeriv fail või kataloog:
- kõigi õigustega,

- mõnede õigustega,
- mitte ühegi õigusega.

Mõnede õigustega fail:
- ainult lugemisõigusega,
- lugemisõiguseta, kuid teiste õigustega.

Mõnede õigustega kataloog:
- on lugemisõigus, kuid mitte sisenemisõigus,
- on sisenemisõigus, kuid mitte lugemisõigus.

See jaotuste nimekiri ei pretendeeri täielikkusele. Praktika on näidanud, et tudengid leiavad ka selliseid mõistlikke jaotusi, mille peale mina pole tulnud.

Sellest pole midagi, kui tudeng pole nii lühidalt või selgelt osanud vastata kui on siinantud näidisvastus. Ka mina mõtlesin selle süsteemi alles näidisvastust koostades välja, see pole mingi klassikaline viis, mille ma oleks loengus vargsi enda teada jätnud.

Mõni minu poolt antud jaotus on tõenäoliselt ebarelevantne, nagu näiteks mõlemad tunnuse "Fail" järgi jagamised, sest normaalne programm, mis nõutud tööd teeb, ei loe tavafaile, vaid ainult katalooge. Aga tont neid programmeerijaid teab, kuidas nad on programmeerinud. Parem karta kui kahetseda. Siiski punkte ei tohi maha võtta selle eest, et niisugused jaotused on puudu.

Hindamine:
- Hinne peaks moodustuma proportsionaalselt ekvivalentsiklasside kaetusele.
- Konkreetsete testide koostamine ei ole tähtis, piisab klasside kirjeldamisest.