Eksamiküsimuste vastused aines Tarkvaratehnika (MTAT.03.094)
Aeg: 16. jaanuar 2007, 09:00 – 13:00

Küsimused

1. (5 p.)
   Küsimus:
   Alustad arendusprojekti projektijuhina. Sinu ülesandeks on arendada arvete ja tellimuste andmevahetussüsteem ühe autokomponentide tootja ja autotootja vahel. Nimeta antud projekti kõige kriitilisemad edutegurid ning riskid. Millised inimesed peaksid olema projekti otsustatavates faasides hõlmatud.

   Vastus:
   - Edutegurid (2 p.):
     - tootjate juhtkonna/otsustajate huvi,
     - kasutajate kaasamine,
     - tootjate IT-osakondade huvi,
     - süsteemi tellijate kooskõlalised huvid/visioon loodavast süsteemist,
     - sobiva arendusmetoodika valik (sobivust määrab hindaja).
   - Riskid (1.5 p.):
     - tootjate juhtkonna/otsustajate mittekaasatus,
     - kasutajate mittekaasatus,
     - tootjate IT-osakondade vastuseis,
     - kaks süsteemi tellijat ebakooskõlaliste huvidega/visiooniga,
     - liidestamine tootjate infosüsteemidega.
   - Inimesed (1.5 p.):
     - tootjate juhtkond/otsustajad,
     - kasutajad,
     - tootjate infosüsteemide tehnilised juhid.
   - Ei loe:
     - tehnoloogilised faktorid.

2. (5 p.)
   Küsimus:
   Nimeta 4 konfiguratsiooni- ja muudatuste juhtimise olulist töövahendit. Kirjelda lühidalt igaühe põhifunktisionaalsust ning millist lisaväärtust nad lisavad arendusprotsessis.

   Vastus:
   - *Version control tool* - kasutatakse nii lähtekoodi kui ka muude projektiga seotud failide hoimiseks ja versioneerimiseks. Võimaldab alati "minna ajas tagasi", ning näha milliseid muutusi on tehtud. Mitme arendusharu toetus.
   - *Build tool* - kasutatakse projekti kompileerimiseks, pakendamiseks jne. võimaldab ilma IDEta deliverable saada (nt. *Ant, Maven, make*).
   - *Continuous integration tool* - kasutatakse regulaarseks projekti ehitamiseks, testide käivitamiseks. Aitab saada kiiret tagasisidet igast projekti tehtud muutusest suvalise arendaja poolt (nt. *CruiseControl*).
   - *Dependencies manager* - kasuatakse projekti kõigi vajalike sõltuvuste kirjeldamiseks, lahendamiseks ning allalaadimiseks (nt. *Ivy, Maven*).

   Hindamine:
   - Iga töövahend annab max 1.25 punkti.
   - Ainult nime eest võiks anda max 0.25 punkti.

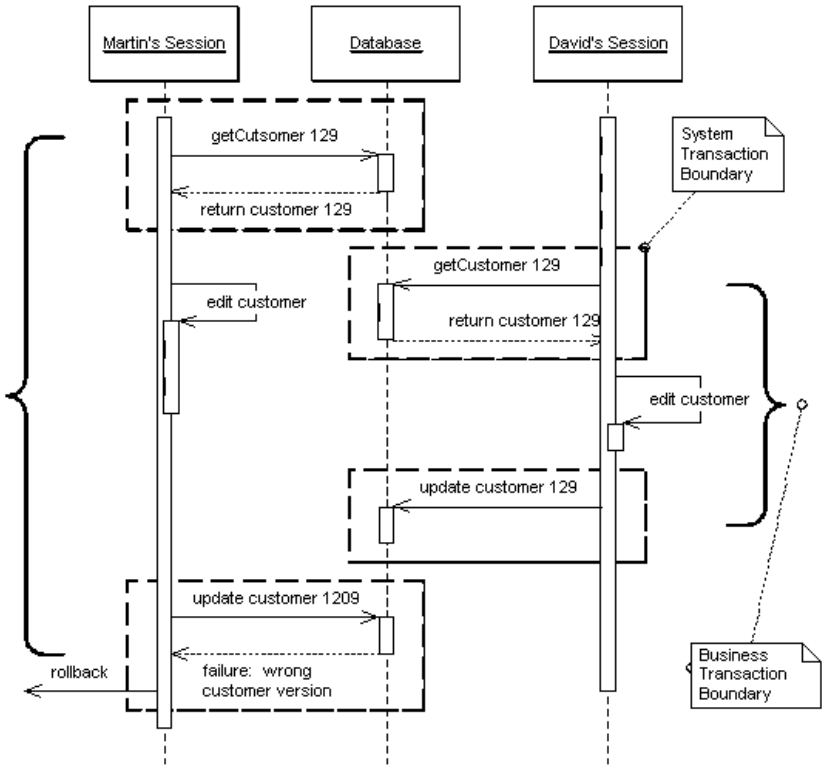
3. (5 p.)
Ülesanne:
Kirjelda lühidalt *optimistic lockingu* toimepõhimõtet. Kirjelda üks võimalus kuidas saab relatsioonilisel andmebaasil realiseerida *optimistic locking* mustrit.

Lahendus:
Optimistic locking muster on lahenduseks olukorrale kus pikkade äritransaktsioonide jooksul kaks või enam kasutajat võivad muuta samu kirjeid ning mingil põhjusel (tüüpiliselt jõudlus) ei saa kasutada andmebaasi või muu transaktsionaalse ressursi poolt pakutavat lukustusmehhanismi.

Lahenduseks lisatakse tüüpiliselt kirjele üks lisatunnus mis on enamasti kas kirje versiooninumber või viimase muutmise kellaaeg. Kirje muutmisel äritransaktsiooni käigus loetakse kirje transaktsioonilisest ressursist koos tema lisatunnusega välisesse seisundiga objekti (nt. HTTPSessioon või Stateful Session Bean). Äritransaktsiooni lõpetamise hetkel võrreldakse transaktsioonilisest ressursis ning välises seisundit hoidvas objektis asuvat lisatunnust. Kui need on samad (mingi teine lõim/süsteem/kasutaja pole vahepeal antud kirjet uuendanud) siis transaktsioon commititakse ning versiooni/ajatemplit keritakse baasis edasi. Kui vahepeal on toimunud muudatus siis tüüpiliselt antakse kasutajale veateade, kuid võib pakkuda ka võimalusi üle kirjutamiseks/mergemiseks.

All on illustreeriv selgitus Martin Fowleri raamatust „Patterns of EAA”.

Hindamine:
- Lahendus peab mainima kas kellaaega või järjest edasi keritava numbrit (50%).
- Tööpõhimõtte kirjeldusel peab olema näha kus hoitakse seisundit mida võrreldakse andmebaasis olnud hetkeseisuga. (50%)



4. (10 p.)
Ülesanne:
Koostada botaanik-amatööri jaoks Eestis leiduvate taimeliikide leiukohtade märkimiseks andmemudel.

Vastavalt järgmisele kirjeldusele luua andmebaasiskeem, kus on vajalikud andmeelemendid jaotatud tabelitesse ja näidatud ära tabelite omavahelised seosed. Elementidel näidata ära andmetüübid ja kohustuslikkus. Seostel näidata ära seose moodustavad väljad. Tabelites näidata ära sisulised võtmed (kui leiduvad).

Taimeliigid on hierarhiliselt süstematiseeritud. Sarnased liigid koondatakse perekondadeks. Sarnased perekonnad sugukondadeks. Edasine grupeerimine pole hetkel tähtis. Samuti pole hetkel olulised need liigid, mida Eesti ei leidu.

Liikidel, perekondadel ja sugukondadel on olemas ametlikud eestikeelsed nimetused ja ka ladinakeelsed. Aga lubame, et ladinakeelsed ei pea botaanik-amatööril andmebaasis alati olemas olema.

SUGUKOND: kukerpuulised (*Berberidaceae*)
PEREKOND: kukerpuu *(Berberis)*
harilik kukerpuu *(Berberis vulgaris)*
SUGUKOND: tulikalised (*Ranunculaceae*)
PEREKOND: käoking (*Aconitum*)
kollane käoking *(Aconitum lycoctonum)*
PEREKOND: siumari (*Actaea*)
salu-siumari *(Actaea spicata)*
PEREKOND: ülane (*Anemone*)
võsaülane *(Anemone nemorosa)*
kollane ülane *(Anemone ranunculoides)*
metsülane *(Anemone sylvestris)*
PEREKOND: varsakabi (*Caltha*)
varsakabi *(Caltha palustris)*
PEREKOND: sinilill (*Hepatica*)
sinilill *(Hepatica nobilis)*

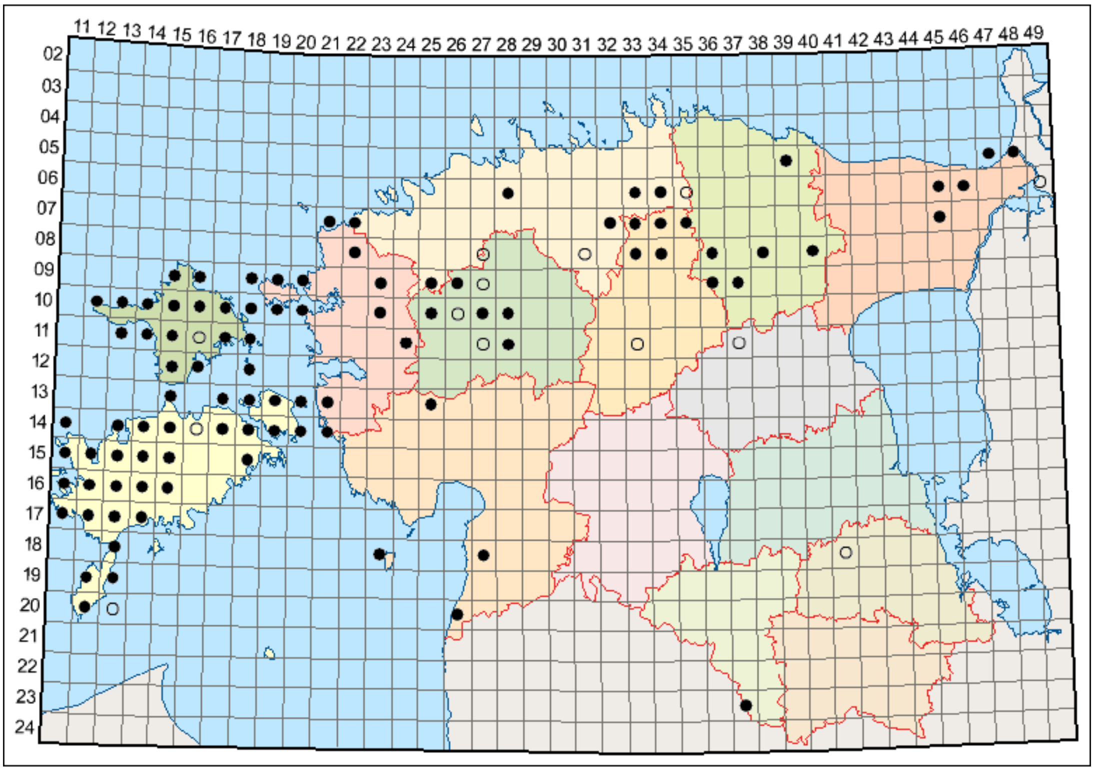
Taimeliikide kohta peab samaa koostada leviaala kaarte. Maa-ala (Eesti riik) on jaotatud umbes 100 km^2 suurusteks ristkülikuteks, mida kutsutakse "ruutudeks" (laius 10 kraadiminutit, kõrgus 6 kraadiminutit). Ruudud on tähistatud kahe numbriga. Y-teljel põhjast lõunasse numbrid 01 kuni 24. X-teljel läänest itta numbrid 11 kuni 49.

Leviala kaardil märgitakse ära liigi leidumine ruudus (täpsemalt pole vaja). Haruldasemate liikide korral tehakse ka vahet "ammuste" ja värskete leidumiste vahel. "Ammu" definitsioon pole hetkel oluline, kuid selle aluseks on leiu vanus. See sunnib leiud varustama vähemalt aastaarvuga. Andmebaasi peab saama kanda ka näiteks 100 aastat tagasi "muistsete" loodusuurijate poolt tehtud leide (need on erialasest kirjandusest kättesaadavad). Igal juhul peab saama leiu juurde märkida leidja nime (või initsiaalid), aga andmebaasi ainukasutaja ei tarvitse ennast märkida. Andmete sisestaja käsutuses peaks olema ka väike märkuse lahter (viide kirjandusele või täpsema asukoha või

taimekoosluse kirjeldus – vastavalt sisestaja kontekstist tulenevale vajadusele).

Looduses tehtud leiu positsioneerimiseks konkreetsesse ruutu saab kasutada teadmist, mis asulad mingites ruutudes asuvad (kusjuures asula võib olla jaotatud mitme ruudu vahel, nt Pärnu-Jaagupi).

Regio teede atlasest võib samuti üles leida ruudu (kraadiminutitel põhinev võrgustik on täpselt sama), kuid identifitseerimiseks vajalikke nimesid ega numbreid atlases märgitud pole. Seega on andmebaasis vaja hoida asulate nimesid, et eri allikatest saadud info kokku viia. Arvestada, et asulate nimed võivad korduda (nt Mõisaküla). Mugava käsitlemise huvides võib andmebaasi (ainu)kasutaja panna ruudule omale meelepärase nime.


11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24


Halinga
Lehtmetsa
Käelase mõis
Salu
Pööravere
Järvesoo
Libatse
Enge j
Enge
Oese
Tühjasma
Lehu
Enge soo
Mõisaküla
Kandi
Eerma
Anelema
Uduvere
Pärnu-Jaagupi
Valistre
Soosalu
Eametsa
2
58°36'

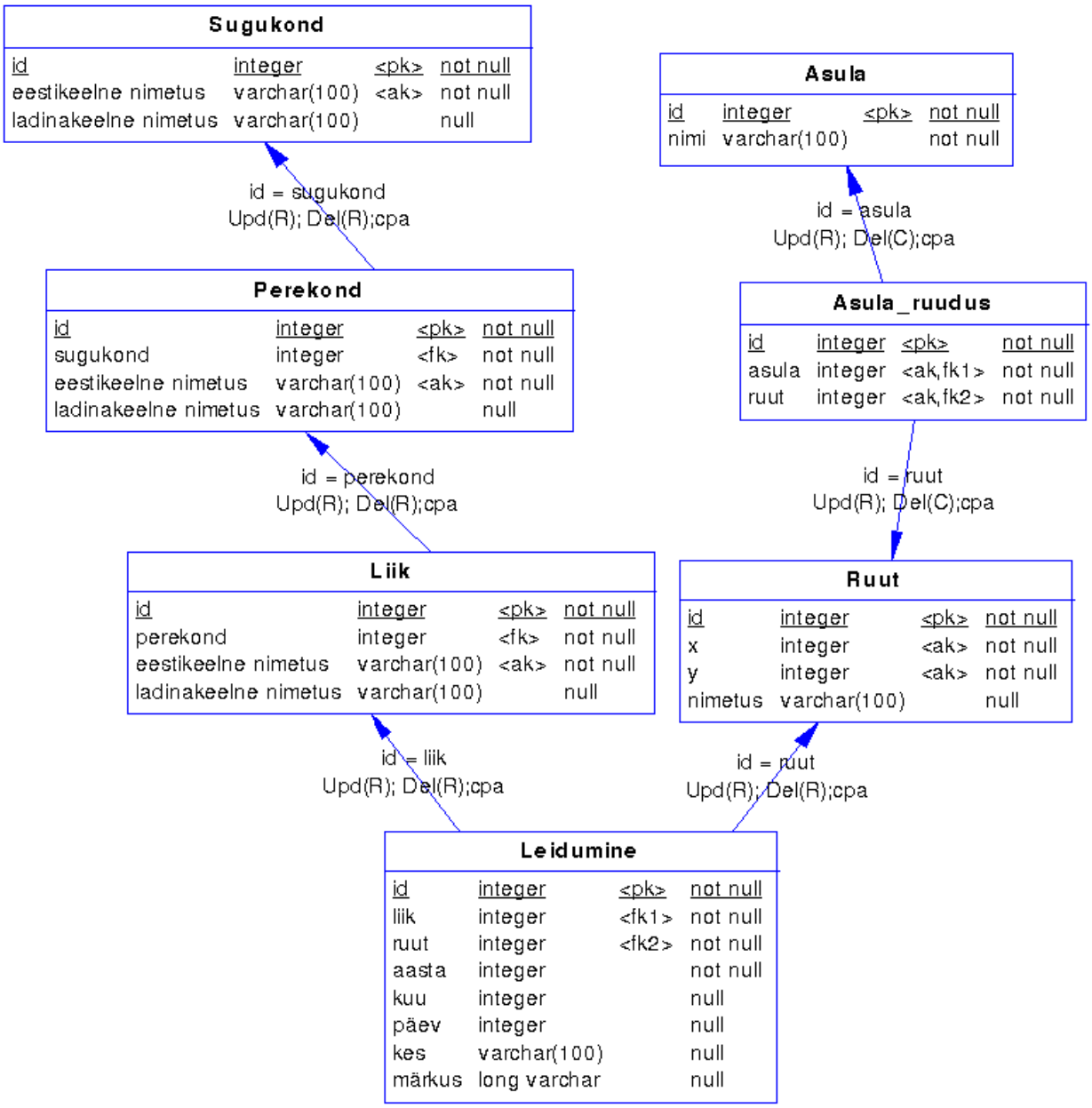
Lahendus:
Võimalik lahendus on allpool.

Võimalikud variatsioonid (optimeerimised) lahenduses

- **Sugukond**, **Perekond** ja **Liik** on üks tabel. Sel juhul vajalik väli, mis näitab ära taseme. Võtmeks oleks sel juhul parem paar (tase+eestikeelne nimi), kuid tegelikult piisab ka nimest.
- **Asula** tabelit pole ja selle asemel on **Asula_ruudus** väli **Asula** tekstitüüpi.
- Tabelis **Ruut** pole välja **Nimetus**. Selle asemel on **asula_ruudus** väli (või kaks), mis määrab asula nime kasutamise ruudu nimes (jah-ei ning järjekord jah korral)
- **Leidumine.Kes** võib olla integer ja seega peab leiduma ka vastav tabel. Aga leidumise tabelis ikkagi mittekohustuslik (NULL).

Hindamine:

- Kõik võimalikud andmed ei ole esitatavad andmemudelis (kuni -5 p.)
- Andmete dubleerimine tabelites/3NF puudumine (kuni -1 p.)
- Valed/puuduvad primaarsed võtmed (kuni -1 p.)
- Valed/puuduvad välisvõtmed (kuni -1 p.)
- Valed/puuduvad väljade tüübid (kuni -1 p.)
- Valed/puuduvad väljade kohustuslikkused (kuni -1 p.)
- Viga oleks NULL väljade panek võtmesse:
  - **Ruut.Nimetus**
  - Ladinakeelsed nimetused
- Sisuline viga oleks märkida võtmeks:
  - **Asula.Nimi**,
  - Tabelis **Leidumine** komplekt: **Liik, Ruut, Aasta**.


Sugukond
id integer <pk> not null
eestikeelne nimetus varchar(100) <ak> not null
ladinakeelne nimetus varchar(100) null
id = sugukond
Upd(R); Del(R);cpa
Asula
id integer <pk> not null
nimi varchar(100) not null
id = asula
Upd(R); Del(C);cpa
Perekond
id integer <pk> not null
sugukond integer <fk> not null
eestikeelne nimetus varchar(100) <ak> not null
ladinakeelne nimetus varchar(100) null
Asula_ruudus
id integer <pk> not null
asula integer <ak,fk1> not null
ruut integer <ak,fk2> not null
id = perekond
Upd(R); Del(R);cpa
id = ruut
Upd(R); Del(C);cpa
Liik
id integer <pk> not null
perekond integer <fk> not null
eestikeelne nimetus varchar(100) <ak> not null
ladinakeelne nimetus varchar(100) null
Ruut
id integer <pk> not null
x integer <ak> not null
y integer <ak> not null
nimetus varchar(100) null
id = liik
Upd(R); Del(R);cpa
id = ruut
Upd(R); Del(R);cpa
Leidumine
id integer <pk> not null
liik integer <fk1> not null
ruut integer <fk2> not null
aasta integer not null
kuu integer null
päev integer null
kes varchar(100) null
märkus long varchar null

5. (10 p.)
Ülesanne:
Alustatakse uue primitiivse tekstriredaktori (a la *Windows Notepad*) arendamist.

Olulised funktsioonid:
- teksti sisestamine,
- tekstis liikumine nii hiire kui klaviatuuri abil,
- teksti *scrolling* (sh keribisribaga),
- automaatne *wordwrap* kui tekst on laiem kui aken,
- teksti osa valimine nii hiire kui klaviatuuri (*Shift*-nooled) abil,
- järgmisele/eelmisele sõnale liikumine (*Ctrl*-nool paremale, *Ctrl*-nool vasakule),
- *cut/copy, paste,*
- *find, find+replace,*
- *load, save, save as.*

Sinu ülesanne on tagada, et toode saaks korralkult automaatsete testidega kaetud.

Unit testing.
- Kirjelda, millistele alamosadele eraldi peaks kirjutama unit testid.
- Millised oleksid tõenäaolised TestCase'de ning nende sees test-meetodite nimed.

Acceptance testing.
- Kirjelda, millist testimise tarkvara oleks tarvis, et seda redaktorit automaatsete testidega katta (see ei pea olema täna reaalselt eksisteeriv toode!).
- Kirjelda pseudokeeles testid, mida teeksid, et kogu funktsionaalsuse olemasolus ja toimimises veenduda.
- Pööra kindlasti tähelepanu veasituatsioonidele ning kasutaja võimalikele veidrustele.

Mõtle loovalt! Ära ole kinni tänases tarkvaras või nähtud meetodites.

Lahendus:
- *Unit testing* - süsteem peaks olema ülesehitatud MVC põhimõttel, mis soodustaks testimist (vt. nt. JTextComponent):
  - *Model* – esitab teksti presentatsioonist sõltumatul kujul ning pakub elementaaroperatsioone teksti osade lugemiseks ja muutmiseks. Ühiktestid peaksid kutsuma iga elementaaroperatsiooni erinevatel erijuhtudel ning testima, et mudeli seis muutus vastavalt. Lisaks peab eraldi testima I/O operatsioone – salvestamist ja laadimist.
  - *View* – tegeleb *Modeli* sisu kuvamisega, peaks sõltuma *Modeli* liidesest, kuid mitte konkreetstest realisatsioonist. See võimaldab testida *View* kasutades lihtsustatud mudeli realisatsioone (nn *mocking*). *View'd* on testida kõige raskem sest väljundiks on graafiline info, mida on raske spetsifitseerida ning raske kontrollida selle vastavust spetsifikatsioonile. Testida saab näiteks teksti *layouti* algoritmi.
  - *Controller* - reageerib UI sündmusteele, muudab *Modelit* ning informeerib *Viewd* olekute muutustest. Peaks sõltuma *Modeli* ning *Viewi* liidestest, et saaks kasutada *mockingut.* Erinevatele UI sündmustele reageerimiseks peaks olema eraldi event handler'id. Iga handlerit peaks

olema võimalik eraldi testida ning veenduda, et ta muudab *Modeli* ning *Viewi* olekut.

Võimalikud *TestCased*:

- *Model*:
  - ParagaphTest: testGetParagraph, testInsertParagraph, testDeleteParagraph
  - TextTest: testGetText, testInsertText, testDeleteText
  - FindTest: findTextSuccess, findTextFail, findTextAtEnd
  - SerializationTest: testWrite, testReadSuccessful, testReadFail
  - ....
- *View*
  - TextLayout: testGetLineNoWrap, testGetLineWithWrap
  - ParagraphLayout: testGetParagraphHeight
  - PositionTest: testGetPositionByXY
  - *...*
- *Controller*:
  - MoveTest: testMoveLeft, testMoveRight, testMoveUp, testMoveDown
  - InsertTest: testInsertChar
  - ParagraphTest: testInsertParagraph
  - DeleteTest: testDeleteNext, testDeletePrevious, testDeleteSelection
  - SelectionTest: testKeySelection, testMouseSelection
  - FindTest: testFindExisting, testFindNotExisting, testFindNext
  - SaveTest: testIllegalFileName, testSuccessfulSave, testFailedWrite
  - LoadTest: testNormalLoad, testNonexistantFile, testBinaryFile
  - ...

- *Acceptance testing* - testimise tarkvara, mida me vajame peaks pakkuma selliseid võimalusi:
  - Variant A: klaviatuuri ja hiire tegevuste ning ekraanipiltide "lindistamine" ning hilisem "maha mängimine" koos erkaanipiltide võrdlusega.
  - Variant B: klaviatuuri ja hiire tegevuste kirjeldamine (näiteks: Type "Mingi tekst", Press Ctrl-Left, Mouse to 10,4, Left-Click, Click Button 'save' jne) ning akna sisu ja failisüsteemi kontrollide kirjeldamine (näiteks: assert textarea contains 'foo bar', assert dialog visible 'save file', assert file exist 'readme', assert text selected 'xyz')

  Võimalikud testid:
  - Normaalne uue dokumendi loomine:
    - start editor
    - type 'tere'
    - press Enter
    - type 'kuku juku'
    - assert textarea contains 'tere\nkuku juku'
    - click button 'save'
    - type '/tmp/test.txt'
    - click button 'ok'
    - assert file exists '/tmp/test.txt' containing 'tere\nkuku juku'
    - close editor
  - Olemasoleva faili redigeerimine:
    - testi jaoks on loodud fail foo.txt sisuga:

      peeter
      meeter
      1223

foo
        - start editor
        - click button 'load'
        - type 'foo.txt'
        - assert textarea contains 'peeter\nmeeter\n1223\nfoo'
        - click button 'replace'
        - type 'eeter'
        - press 'Tab'
        - type 'auk'
        - click button 'replace all'
        - assert textarea contains 'pauk\nmauk\n1223\nfoo'
        - click button 'save'
        - close editor
        - assert file exists 'foo.txt' containing 'pauk\nmauk\n1223\nfoo'
    - Find/replace tühjas failis
    - Olematu faili avamine
    - Sulgemine enne salvestamist
    - Järgmisele sõnale hüppamise klahv teksti lõpus
    - ...

Hindamine:

- *Unit testing*
    - MVC mudeli kirjeldamine (kuni 2 p.)
    - TestCase'd proportsionaalselt antud näidete kaetusele (kuni 3 p.)
- *Acceptance testing*
    - Sobiva testimistarkvara kirjeldus (kuni 2 p.)
    - Funktsionaalsuse ning eriolukordade kaetus testidega vastavalt kontrollija hinnangule (kuni 3 p.)