# Rühmatöö: Kes tahab saada miljonäriks?!

Autorid: Kärt Miidla ja Priit Pärn

Projektiks tegime mälumängu tuntud telesaate “Kes tahab saada miljonäriks?” järgi, kus programm küsib kasutajalt erinevaid küsimusi ning kasutaja peab küsimustele vastama.

projekti põhjalik kirjeldus, kus on kirjas programmi eesmärk ja selgitus programmi üldisest tööst, vajadusel lühike kasutusjuhis;

Küsimused mõtlesime ise välja või valisime välja raamatust “Kes Tahab Saada Miljonäriks?”. Ühes mängus on kokku 15 küsimust ning meie koostasime 2x15 küsimuste *set*’i. Programmi käivitamisel hakkab taustal mängima muusika. Esimesena avaneb aken koos tutvustava tekstiga, kus saab ka valida, millise set’i küsimustega tahetakse mängida, kui mängija ei vali ühtegi tulevad esimese set’i küsimused. Tahetakse mängija nime ja kõige alumisele nupule vajutades tuleb uus aken koos küsimustega. Kõige esimesena saab vastata esimesele küsimusele, kui sellele on vastatud avaneb teine küsimus ja nii lõpuni välja. Küsimustele vastamine toimub eraldi aknas, kus vastusevariandid on suvalises järjekorras. Mängijal on 60 sekundit aega, et küsimusele vastata, muidu langeb ta automaatselt mängust välja(mängijale on näha palju tal veel aega on) .Mängijal on võimalik 3 korda kasutada vihjevõimalust, kus ekraanile kuvatakse küsimusega seotud vihje. Kui mängija vastab õigesti, siis tuleb vastav teade. Kui mängija vastab kõigile 15. küsimusele õigesti on ta võitnud. Programmi eesmärk on laiendada mängija silmaringi.

iga klassi kohta eraldi selle eesmärk ja olulisemad meetodid;

Programmis on 3 klassi:

* KesTahabSaadaMiljonäriksV2 – põhiklass, kus toimub „mängu mängimine“.
  + Meetod: start(pealava), kus toimub graafiliste lisandite realiseerimine.
  + Meetod: music() paneb muusika mängima programmi taustale.
  + Meetod: küsimus(küsimus1, vv1, vv2, jne) koostab uue akna küsimusega ja sellele küsimusele vastavate valikvastustega.
  + Meetod: failiKirjutamine(kirjutatav) kirjutatakse faili mängija nimi ja tema valitud vastusevariandid.
  + Meetod: küsimusteAndmed(failiteekond) loeb failist info ja salvestab selle.
  + Meetod: randomVastused(variandid) lööb vastusevariandid segi.
* Andmed – klass, kust saadakse andmed.
  + Konstruktor.
  + Get - meetodid.
* Mängija – klass, kus luuakse mängu mängija.
  + Konstruktor.
  + Get - meetodid
  + Set - meetodid

projekti tegemise protsessi kirjeldus (erinevad etapid ja rühmaliikmete osalemine neis);

1. Eelmisele projektile graafilise kasutajaliidese realiseerimine – Priit
2. Kõigi akente(küsimused, vastamine, vastavad teated) ja hiire ning klaviatuuri tekitatud sündmuste töötlemine – Priit
3. Faili kirjutamine – Priit
4. Kogu disaini tegemine – Kärt
5. Erinditöötlus - Priit

iga rühmaliikme panus (sh tehtud klassid/meetodid) ja ajakulu (orienteeruvalt);

Priit – tegi meetodid: start, music, küsimus, failiKirjutamine, küsimusteAndmed. Ajakulu: 22h.

Kärt – tegi küsimuste set’id ning meetodid: randomVastused.

tegemise mured (nt millistest teadmistest/oskustest tundsite projekti tegemisel puudust);

Kärt – Erinevate klasside kasutus tekitas segadust, kuid muutis koodi selgemaks. Siiamaani ei saa ma aru Get meetodi ja konstruktori vajalikkusest. Samuti tekitas palju probleeme alguses java failide kompileerimine (kood ei läinud käima), kuid need probleemid said lahedatud. Algselt kasutasin GitHub Desktopi, kuid ükshetk (settingute muutmisest) näitas GH Desktop nii palju tehtud muutmisi (27000+), et laadisin edaspidi manuaalselt programme Git’i üles.

Priit – Kõige suurem raskus oli graafilise liidese realiseerimine, sest nii suure asjaga polnud veel tegelenud ja gradle-ga polnud veel palju kogemusi.

hinnang oma töö lõpptulemusele (millega saite hästi hakkama ja mis vajab arendamist);

Vahva programm, mida me mõlemad oleme juba oma pereliikmete peal katsetanud. Kogu asjale graafilise liidese realiseerimine oli raske, aga võib öelda, et tulemus on ilus. Arenguruumi veel on, näiteks osad aknad oleks võinud paremini teha.

selgitus ja/või näited, kuidas programmi osi eraldi ja programmi tervikuna testisite ehk kuidas veendusite, et programm töötab korrektselt.

Testisime iseseisvalt enne GitHubi laadimist, kuigi vahepeal jäi sisse apsakaid, mida kohe ei märganud. Enne juhendajale esitamist kontrollisime mõlemad eraldi programmi ja katsetasime kõikvõimalikke „eriolukordi“. Oli arvutis olemas test fail, kuhu sai asju tõsta just spetsiaalselt nende testimiseks.