Trips Traps Trull

Uku Kert Paidra, Raido Maasik

# Kirjeldus

Tegemist on elementaarse mänguga Trips Traps Trull, kus eesmärgiks on saada horisontaalselt, vertikaalselt või diagonaalselt ritta kolm samasugust märki. Mängija mängib arvuti vastu ning mängu alustaja valitakse juhuslikult. X alustab alati, aga kumb kumma saab, on juhuslik. Mäng lõppeb kui kumbi mängijatest on saanud ritta kolm enda märki või kui mängulauale ei saa enam käike teha, mille korral mäng kuulutab välja viigi.

# Kasutajajuhis

Mängu mängimiseks tuleb avada fail kas IntelliJ’s, Eclipses või cmd’s. Mängu käivitamiselt käsurealt on soovitatav muuta käsurea kodeering UTF-8 peale. Muidu ei joonistu kasti jooned korralikult välja(Seda vähemalt Windowsi puhul). Seejärel joonistub ekraanile mänguväli ja oma valiku saab teha numbriklahvidega 1-9. NB! Silmas tuleb pidada, et numbrid jooksevad vasakult paremale, ülevalt alla ehk nagu telefoni klahvistik ja mitte arvuti numpad. Mängu lõppemisel program sulgeb iseennast.

# Klassid

Class Player on mõeldud mängijate loomiseks. Selle puhul loob Game peaklass kaks mängijat “computer” ja “human” ning omistab neile kummalegi märgi(CROSS või NOUGHT). Samuti luuakse kaks klass Player isendit player1 ja player2. See järel omistatakse player1 see mängija, kumma mark on CROSS ja player2 omistatakse teine mängija. Klassis muid meetodeid peale kahe konstruktori, ühe kloonija ning getterite ja setterite ei ole.

Mänguvälja ehitusklotsiks on Class Space, mis on isendiklass kelle ainuke omadus on märk, ehk CROSS või NOUGHT või EMPTY. Selle väärtuse jaoks on enum nende väärtustega. Nii on see paremini loetav kui numbriväärtuste kasutamine. Peamine huvitav meetod on staatiline buildEmptySpace(), mis tagastab uue EMPTY väärtusega välja ja on ainuke viis, et saada uus väli, ehk asendab constructori. Põhjuseks on selle parem loetavus. Muidugi on välja märgi getter/setter ja toString.

Mänguvälja ehitab Class Board, isendiklaasina on tal omadusena kahe dimensionaalne Space isendi masiiv. Sellel klassil samuti puudub traditsiooniline constructor ja see on asentatud staatilise meetodi buildEmptyBoard() poolt, ning ta tagastab uue Board isendi, koos täidetud masiiviga. Nii on võimatu saada mänguvälja, mis ei oleks sobilik mängu jaoks. Muidugi on get/set meetodid, mis võtavad välja numbri ja märgi(set meetod), selgitab lihtsalt välja 2 dimensionaalse diagonaali kordinaadi ja tagastab/muudab seda. Lisaks on get meetotit utiliseeriv checkIfSpaceEmpty ja hasWon, mis kontrollib kas antud märk on võitnud.

Mängu isendiklaasi on Class Game , kelle omadusteks on Board, Scanner sisestuseks ja Random arvutimängija juhusliku valiku jaoks. Tähtsaim meetod on playGame(), mis viib läbi mängu käigud ja joonistab kasutajatagasiside. Selle läbiviimiseks on ülejäänud meetodid mängu käigu osad: valiku sisestuse küsimine/edastamine, vastava märgi muutmine jne.

Viimakse Class Main , kes oma main meetodis teeb uue Game isendi ning paneb ta mängima.

# Tegevus

Alustasime idee brainstormimisega, mille katkestas viiruse uudistega seotud rutiinimuutusega. Teemale tagasitulles otsustasime trips traps trulli kasuks. Kiire aruteluga panime paika, mis osad keegi teeb. Raido võttis mängu põhiloogika ja disaini, ning Uku võidukontrolli ja kasutajale nähtava osa. Raido kirjutas enda osa ja Uku lisas oma osa, edasi lihvimine käis edasi tagasi. Projekti tegemise aeg mahub nädala sisse.

# Panused

Raido:Ma ehitasin põhimängulookiga, millele Uku saaks oma osad lisada. Tegin Space , Mark , Board ja algloogika Game klassi. Neid redigeeris/muutis isendiks Uku. Edasi olid ainult väiksed lihvimised.

Tööaeg ligikaudu 4 h algse versiooni jaoks ja 2 h kontrollide, augute katmise ning poleerimiseks.

Uku: Raido ehitatud osale lisasin mina juurde kontrolli, kas mäng on võidetud või viigistatud. Lisasin Player klassi ja muutsin Game klassi, et see võtaks andmeid Player isenditelt. Samuti kirjutasin pea mänguloopi lihtsamaks, peamiselt Player klassi lisamisega, kuid ka turn() meetodi lisamisega(üldistus Raido kirjutatud koodist) ja extractimisega. Tegin ka mängu alguse juhuslikuks, et X alustaks alati. Samuti kirjutasin koodiosa, mis väljastab ekraanile ruudustiku ja mängu seisu ja mängu lõppemise korral ka sõnumi võitjale.

Tööaeg ligikaudu 5h 30min.

### Uku kirjutatud klassid ja meetodid:

Class Player terviklikult

Class Board meetod haswon()

Klassi Game meetod checkIfWin()

meetod turn() (võttes aluseks Raido kirjutatud käigusüsteemi)

meetod drawBoard()

meetod markToString()

# Tegemise mured

Uku: Ma ei tundnud väga millestki puudust. Kõik murekohad olid lahendatavad kiire Google’I otsinguga.

Raido: põhimõttekohad olid seotud, üldise ülesehitusega ja milline see peaks reaalses kasutuses välja nägema. Võibolla üldine töökäik võiks olla selgem, git selgem ja testimine tuleb arsenali lisada.

# Hinnang töö lõpptulemusele

Arvan, et lõpptulemus on hea. Mängu saab mängida ilma eelteadmisteta ja programm on sujuvalt jooksnud kõik testimise korrad. Tunnen, et programm on piisavalt kommenteeritud ja piisavalt lihtsalt kirjutatud, et sellest saaks kolmas koodija vähese pealevaatamisega juba aru. Kui oleks vaja programmi edasi arendada, saaks koodida arvutile mingi elementaarse AI, millega ta ei tee enam juhuslikult valikuid, vaid üritab võita.

# Testimine

Testisime programmi selle kasutamisega, et kontrollida, kas (peaaegu) igas mänguseisus program töötab, joonistab seisu korrektselt ja kuulutab viigid, võidud ja kaotused korrektselt. Samuti, et progamm sulgeks ennast pärast mängu lõppu.