1. Henkilötiedot

Miia Konu
101816559
Tietotekniikka, ensimmäinen vuosi
4.5.2024

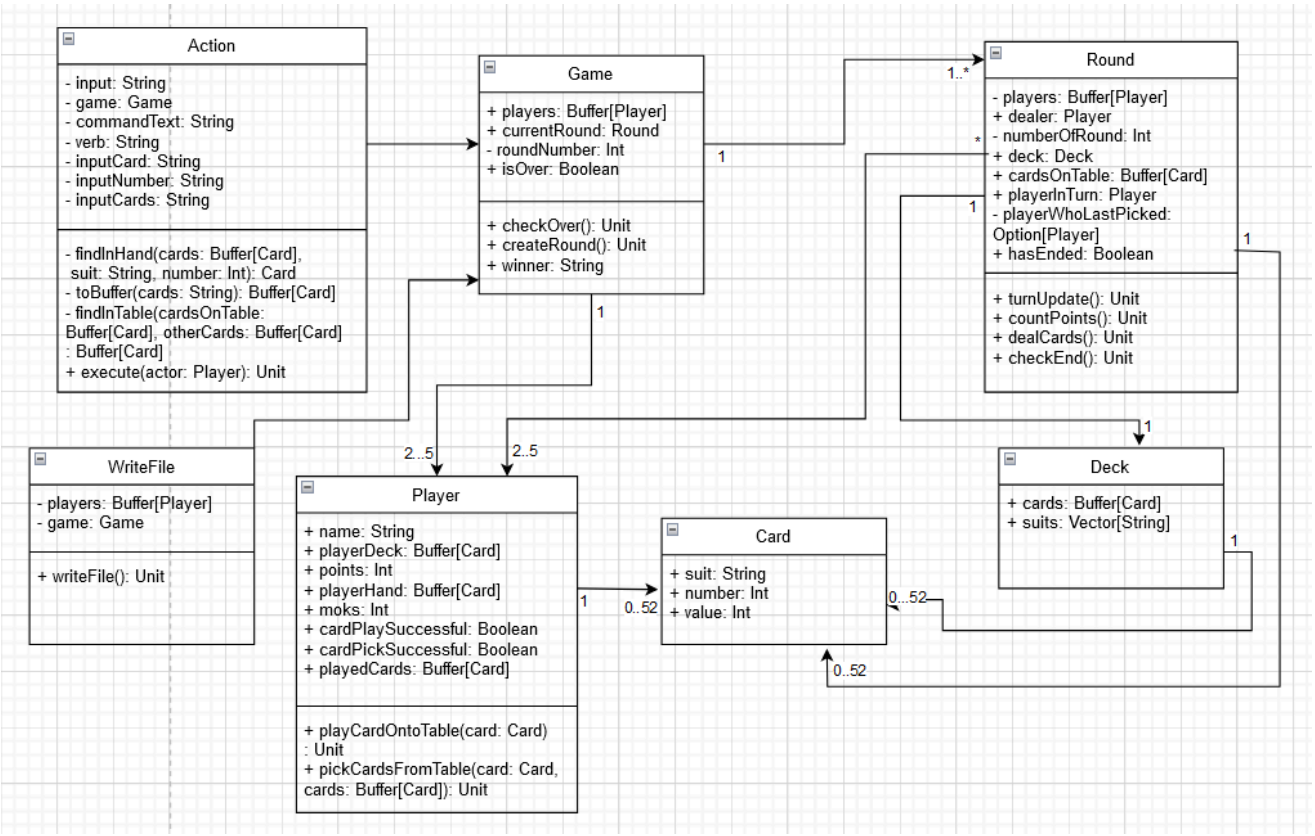
2. Yleiskuvaus

Tein merkkipohjaisen kasino-pelin keskivaikealla vaativuustasolla. Pelissä voi olla 1-5 pelaajaa kerrallaan, ja konsolissa näkyy aina vuorossa olevan pelaajan kortit ja pöydässä olevat kortit. Pelaaja voi play-komennolla pelata jonkin korttinsa pöytään, ja pick-komennolla nostaa pöydästä kortteja omaan pakkaansa. Ohjelma tarkistaa, ovatko pelaajan tekemät siirrot mahdollisia (onko pelaajalla hänen pelaamaansa korttia kädessä, onko hänen nostamiaan kortteja pöydässä, voiko pöydästä nostaa tietyt kortit pelaajan ilmaisemalla kortilla). Kierrosten välissä ohjelma printtaa pelaajien keräämät pakat ja kertoo montako mökkiä pelaaja keräsi, sekä kertoo jokaisen pelaajan pisteet (eroaa suunnitelmasta, jossa piti printata vain pisteet). Pisteet lasketaan kierrosten välissä. Peli loppuu, kun jokin pelaaja kerää yli 16 pistettä.

3. Käyttöohje

Ohjelma käynnistetään Main.scala tiedoston avulla. Pelin alussa pelaajan tulee kertoa pelaajien nimet pilkulla erotettuna (esim. Tiina, Jaana). Peliä pelataan komentojen avulla, jossa pelaaja voi play- ja pick-komennoilla pelata ja nostaa kortteja. Kortteja pelatessa tulee play-komennon jälkeen kirjoittaa pelattavan kortin maa ja numero. Muutoksena suunnitelmaan pelaaja ei näytäkään komennolla omia korttejaan, vaan ne ovat näkyvillä heti vuoron alussa. Pick-komentoa käytettäessä tulee kirjoittaa komennon jälkeen se kortti, jolla haluaa nostaa pöydästä kortteja, ja tämän jälkeen puolipiste, ja sitten kirjoittaa nostetut kortit pilkulla erotellen. Esim. Jos haluaa pata 10 -kortilla nostaa pöydästä ruutu 5 ja pata 5 -kortit, sen voi ilmaista näin: pick spade 10; club 5, spade 5. Pelin voi lopettaa kesken komennolla q!. Jos kortilla on kädessä eri arvo kuin pöydässä, sen kädessä oleva arvo on kortin nimen jälkeen merkitty sulkujen sisään. Pöydässä korttien arvo on vain kortin numero. Korttia pelatessa ei tarvitse kirjoittaa kortin arvoa sen perään (eli riittää vain heart 1).

4. Ohjelman rakenne



- Game

Game-luokassa on kaikki pelin kulkuun liittyvät toiminnot, siellä esim. on tallessa kaikki pelaajat, pidetään kirjaa meneillään olevasta kierroksesta, voidaan luoda uusia kierroksia, tarkistetaan pelin lopussa pelin voittaja ja tarkistetaan, onko peli jo loppunut. Keskeisimmät metodit ovat createRound, joka luo uuden kierroksen ja tyhjentää pelaajien pakat ja mökit ja checkOver, joka tarkistaa, onko yhdelläkään pelaajalla 16 pistettä tai enemmän.

- Round

Round-luokassa on pelin yksittäisiin kierroksiin liittyvät toiminnot. Siellä pidetään kirjaa pöydässä olevista korteista, jaetaan kierroksen alussa kortit, päivitetään jokaisen pelaajan vuoron jälkeen pakka, pelaajan käsikortit, pöydällä olevat kortit ja vuorossa oleva pelaaja. Keskeisimmät metodit: dealCards, joka jakaa jokaiselle pelaajalle neljä korttia, turnUpdate, joka päivittää pöydän kortit, pakan, pelaajan kerätyt kortit ja pelaajien käsikortit, sekä siirtää vuoron seuraavalle pelaajalle, jos pelaajan siirto (kortin pelaus tai kortilla nosto) oli laillinen ja countPoints, joka jakaa jokaiselle pelaajalle pisteet.

- Player

Player-luokassa on jokaista yksittäistä pelaajaa koskevat metodit. Siellä pidetään kirjaa pelaajan käsikorteista, kerätystä pakasta, pisteistä ja mökeistä (koodissa “moks” koska

kuulosti hauskalta :D). Player-luokassa on myös metodit kortin pelaamiselle (playCardOntoTable) ja korttien nostamista varten (pickCardsFromTable), ja siellä myöskin tarkistetaan, onko pelaajan siirto mahdollinen.

- Deck

Deck-luokassa on tallessa pakassa olevat kortit (Card-oliot). Kun pakka luodaan, sen kortit shufflataan, jotta jokainen pakka on erilainen.

- Card

Kuvaa pelissä olevia kortteja. Card-luokassa on erillinen value-muuttuja, sillä joillain korteilla on kädessä eri arvo kuin pöydässä.

- Action

Action-luokassa “muunnetaan” käyttäjän komennot metodeille sopivaan muotoon eri metodien avulla, ja kutsutaan pelaajan metodeita pelaajan komennoista riippuen. Metodi findInHand, toBuffer ja findInTable muuttavat käyttäjän inputin niin, että niitä voidaan käyttää metodissa execute. Metodi execute kutsuu tietyllä komennolla tiettyä pelaajan metodia.

- WriteFile

WriteFile-luokan writeFile-metodi kirjoittaa pelin tiedot pelin lopuksi gamedata tiedostoon.

- Exceptions

InvalidDataException heitetään aina, kun käyttäjän antama input ei ole oikeanlainen.

- Main

Ohjelma ajetaan Main-tiedoston kautta. Mainissa on pelin pelattavuuden kannalta oleelliset asiat, eli se printtaa tiettyjä asioita pelin vaiheesta riippuen.

5. Algoritmit

Keskeisin algoritmi on korttien tarkistusalgoritmi, joka tarkistaa, voiko jollain kortilla nostaa pöydästä pelaajan haluamia kortteja. Algoritmi tarkistaa ensiksi onko pelaajalla kädessä se kortti, jolla hän yrittää nostaa, ja onko korteissa, jotka hän yrittää nostaa duplikaatteja. Tämän jälkeen:

Jos nostettavia kortteja on yksi, ohjelma vertaa pelatun kortin arvoa ja nostetun kortin lukua toisiinsa, ja jos ne ovat samat, molemmat kortit lisätään pelaajan pakkaan.

Jos kortteja on enemmän kuin yksi, kortit käydään yksi kerrallaan läpi siten, että ensiksi verrataan yksittäisen kortin numeroa nostetun kortin arvoon, ja jos ne ovat samat, nostettu kortti lisätään täsmäävien korttien kokoelmaan. Tämän jälkeen kortteja lisätään yksi kerrallaan toisiinsa, ja verrataan tätä arvoa pelatun kortin arvoon, ja jos arvot ovat samat,

kaikki summassa mukana olevat kortit lisätään mukaan täsmäävien korttien kokoelmaan. Näin, kun kaikki kortit käydään läpi, myös niiden kaikki kombinaatiot käydään läpi. Lopulta verrataan täsmäävien korttien kokoelmaa nostettujen korttien kokoelmaan, ja jos niissä on samat kortit, nosto on onnistunut.

6. Tietorakenteet

Käytän ohjelmassani suurimmaksi osaksi kokoelmatyyppeinä muuttuvatilaisia buffereita, sillä ohjelmassa on paljon tietoja, jotka päivittyvät jatkuvasti. Esimerkiksi korttipakasta lähtee kokoajan kortteja, pelaajien käsikortit vaihtuvat, pöydästä lähtee ja siihen lisätään kortteja, joten muuttuvatilaiset kokoelmatyypit olivat pelissäni paljon toimivampia. Kokoelmiin lisätään jatkuvasti alkioita (peliin pelin alussa pelaajat, korttipakkaan kortit, pelaajien käsiin kortit) ja niistä myöskin poistetaan kortteja (kierrosten välissä pakka, pelaajien pakat ja pelaajien pelatut kortit tyhjennetään, ja pakasta ja pöydästä poistetaan kortteja pelaajien niitä nostaessa).

7. Tiedostot ja verkossa oleva tieto

Ohjelmani tallentaa pelin lopuksi pelitilanteen tekstitiedostoon, jossa kerrotaan pelin päiväys, pelaajien nimet, heidän pisteet ja voittaja. Suunnitelmasta erosin siten, että tallennankin tiedot vasta pelin lopuksi, enkä kesken pelin, sillä halusin tallentaa vain sen mikä tilanne pelissä oli sen loputtua.

8. Testaus

Testasin ohjelmaani aluksi tekemällä printtaavan funktion, jonka avulla testasin korttien tarkistusta, pakan jakamista, kortin pelaamista ja korttien nostamista pöydästä. Siirryin kuitenkin nopeasti testaamaan peliä komentorivin kautta, ja pelaamaan sitä läpi, lisäten samalla ominaisuuksia jotka puuttuivat pelistä, tai olivat virheellisiä. Koin tämän paremmaksi kuin testiluokkien rakentamisen, sillä peliä pelatessa virheet näkyvät suhteellisen selkeästi ja ovat helposti löydettävissä. Testaisin aina myöskin heti ohjelman muokkauksen jälkeen, jolloin virheet oli helppo paikantaa.

9. Ohjelman tunnetut puutteet ja viat

Välillä jos käyttäjä kirjoittaa komennon väärin, ohjelma printtaa kaksi eri vastausta tähän (toisaalta tämä ei ole niinkään puute tai vika, enemmänkin johtuu siitä, että lisäsin

poikkeukset myöhemmin kuin muut inputin tarkistavat kohdat). En ole ainakaan havainnut suuria puutteita ohjelmassa, mutta yleisesti koodi on huonolaatuista ja toistaa paljon itseään, ja esim Action-luokan metodit olisi varmasti voinut tehdä paljon tehokkaammin. Ohjelmassa myös vaaditaan käyttäjältä tarkkaavaisuutta, jotta hän kirjoittaa komennot ja kortit oikein. Jos peliä pelaisi oikeasti, olisi myöskin huono, että näkee edellisen pelaajan kortit. Yksikkötestit olisi myöskin pitänyt tehdä.

10. 3 parasta ja 3 heikointa kohtaa

Olen tyytyväinen miten sain lopulta käyttäjän komennot toimimaan oikein, ja esimerkiksi aluksi minulla oli vaikeuksia yhdistää kortteja jotka pelaaja kirjoittaa inputissaan, esim. pöydällä oleviin kortteihin, sillä ohjelma ei nähnyt kahta eri Card-oliota, joilla on sama maa ja sama numero, samana oliona. Siksi kehitin muutamat metodit, jotka “yhdistivät” ne samaksi. Muutenkin esim. tarkistusmetodi tuntuu toimivan hyvin. Yksi suurimmista heikoista kohdista on jo aikaisemmassa kohdassa mainittu koodin laatu. Koodi on aika toisteista ja if-lause painotteista :D ja esim. poikkeukset ovat sekavia, ja ne olisi voitu tehdä paljon selkeämmin ja tehokkaammin (tällöin olisi myöskin helpompi korjata virheitä). Monet metodit ovat myöskin pitkiä ja sekavia (esim. pickCardFromTable), joten niitä olisi voinut jakaa pienemmiksi koodia kirjoittaessa. Myöskin esim. main-funktio olisi kannattanut jakaa pienemmiksi apufunktioiksi.

11. Poikkeamat suunnitelmasta, toteutunut työjärjestys ja aikataulu

Toteutunut työjärjestys oli suurinpiirtein sama kuin suunnitelmassa, joskin jotkin osa-alueet menivät enemmän päällekkäin kuin suunnitelmassa. Ohjelman testaaminen oli käyttöliittymän kautta helpointa, joten aloin rakentamaan sitä jo heti kun olin tehnyt ensimmäiset korttien pelaus -metodit. Tein ohjelman viimeiset sisäiset rakenteet, kuten pisteiden lasku ja kierrosten vaihtuminen, sen jälkeen, kun olin tehnyt Action-luokasta toimivan ja pelistä pelattavat. Aika-arviot pitivät suurimmaksi osaksi paikkaansa, vaikkakin joihinkin asioihin käytin enemmän aikaa, ja joihinkin vähemmän. Esimerkiksi päädyin tekemään tiedostoon tallentamisesta todella yksinkertaista, joten siihen ei mennyt paljoa aikaa, ja käytin sitä aikaa enemmän käyttöliittymään ja korttien tarkastus metodien hiomiseen, ja ohjelman testaukseen. Olin koko projektin ajan myöhässä suunnitelmastani, ja opinkin projektin aikana, että miksi kannattaa aloittaa ajoissa. Projektissa oli monia asioita,

joita olisin halunnut kehittää pidemmälle, mutta en halunnut priorisoida niitä deadlinen lähestyessä.

12. Kokonaisarvio lopputuloksesta

Lopputulos on mielestäni hyvä, ja vaikka koodissa ja ohjelmassa onkin paljon parannettavaa, koen että opin projektia tehdessä paljon ja pidin hauskaa :) ja sain siihen lisättyä kaikki tärkeimmät ominaisuudet. Haluaisin erityisesti kehittää tiedostoon tallentamista, sillä se jäi projektissani aika pieneksi osaksi koko projektia. Jos alottaisin projektin alusta, pyrkisin rakentamaan koodistani selkeämpää, ja tekemään siitä laadukkaampaa.

13. Viitteet ja muualta otettu koodi

Katsoin mallia Action-luokkaan o1-kurssin materiaaleista (tekstipelistä).