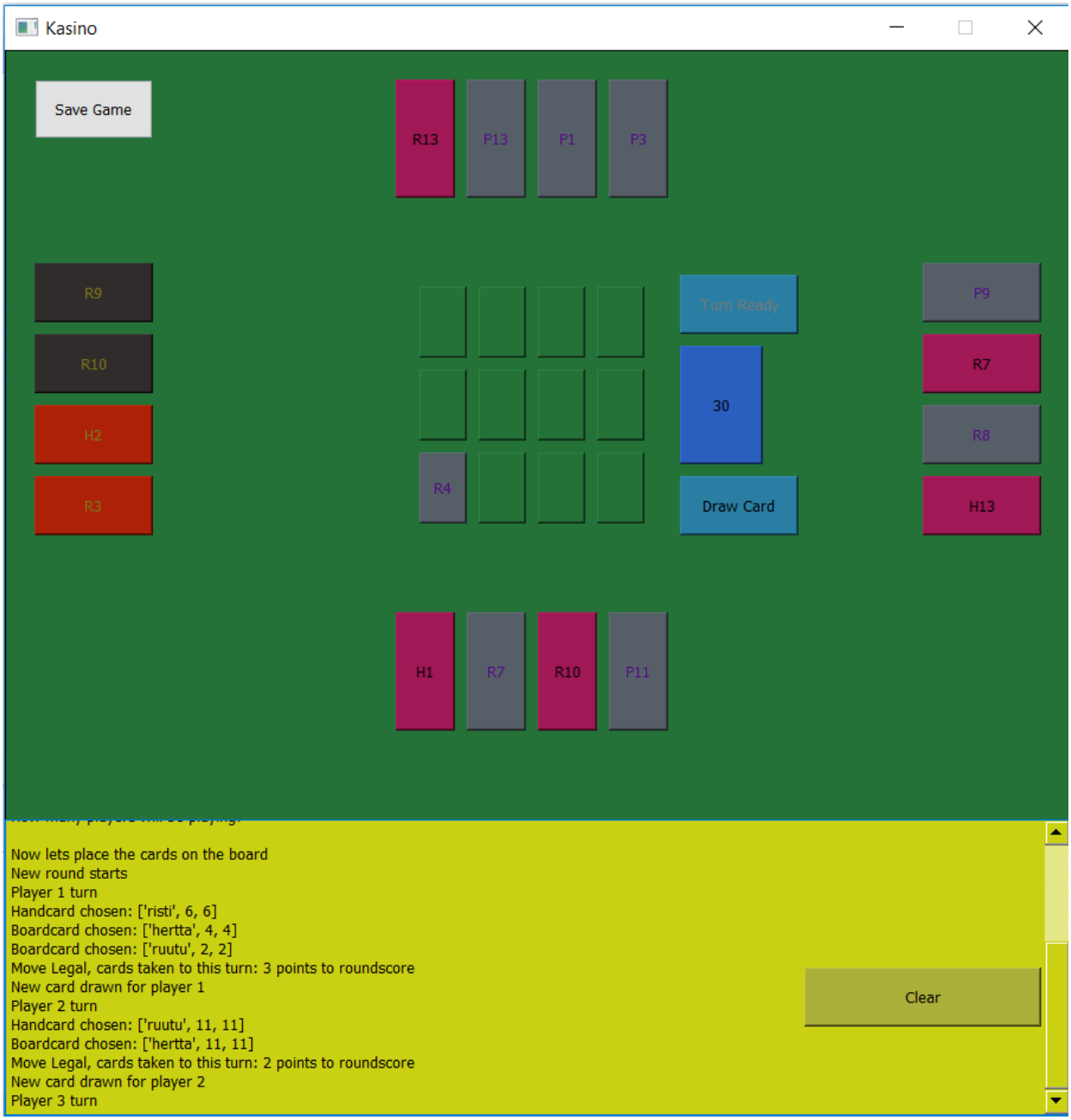
Kasinon Dokumentti

Lari Haapaniemi

651624

Informaatioteknologia

21.4.2019



Yleiskuvaus

Projektin aiheena oli luoda korttipeli, joka noudattaa yleissuunnitelmassa annettuja sääntöjä. Tässä vielä kertaus säännöistä, jotka ovat pysyneet samana muutamaan poikkeusta lukuun ottamatta, jotka kirjaan oikeille paikoille säännöissä, sekä kerron niiden jälkeen vielä tehtävänannosta poikkeavat piirteet.

Mitä kukin pelaaja tekee vuorollaan

Pelaaja voi kerrallaan käyttää jonkin kädessään olevista korteista: joko ottaa sillä pöydästä kortteja tai laittaa kortin pöytään. Jos pelaaja ei voi ottaa mitään pöydästä täytyy hänen laittaa jokin korteistaan pöytään, tai vaihtoehtoisesti kaikkien korttien ollessa käytössä pelaaja voi skipata vuoronsa.

Jos pelaaja ottaa pöydästä kortteja hän kerää ne itselleen pinoon. Pinon sisällöstä lasketaan korttien loputtua pisteet.

Pöydässä olevien korttien määrä voi vaihdella vapaasti kahteentoista korttiin asti. Jos joku vaikkapa ottaa kaikki kortit, täytyy seuraavan laittaa jokin korteistaan tyhjään pöytään.

Aina käytettyään kortin, pelaaja ottaa käteensä pakasta uuden, niin että kädessä on aina 4 korttia. (Kun pöydällä oleva pakka loppuu, ei oteta enää lisää vaan pelataan niin kauan kuin kenelläkään on kortteja kädessä. Tällöin siis tietenkin alle 4 korttia on mahdollinen tilanne.)

Kortilla voi ottaa pöydästä yhden tai useampia samanarvoisia kortteja ja kortteja, joiden summa on yhtä suuri, kuin kortin jolla otetaan.

Jos joku saa kerralla pöydästä kaikki kortit kerralla, hän saa ns. mökin, joka merkitään muistiin.

Erikoiskortit

Pelissä on muutama kortti, joiden arvo kädessä on arvokkaampi kuin pöydässä,

Ässät : kädessä 14, pöydässä 1

Ruutu-10: kädessä 16, pöydässä 10

Pata-2: kädessä 15, pöydässä 2

Muiden korttien arvot ovat samat sekä pöydässä että kädessä.

Pistelasku

Kun kaikilta loppuvat kädestä kortit saa viimeksi pöydästä kortteja ottanut loput pöydässä olevat kortit. Tämän jälkeen lasketaan pisteet ja lisätään ne entisiin pisteisiin.

Seuraavista asioista saa pisteitä:

Jokaisesta mökistä saa yhden pisteen

Jokaisesta ässästä saa yhden pisteen

Eniten kortteja saanut saa yhden pisteen

Eniten patoja saanut saa 2 pistettä

Ruutu-10 kortin omistaja saa 2 pistettä

Pata-2 kortin omistaja saa pisteen

Eli yleisiin sääntöihin on tullut kaksi muutosta. Pöydällä voi olla maksimissaan 12 korttia, johtuen graafisen liittymän rajoitteista, sekä lisäsin vuoron skippaamistoiminnon pakan ollessa tyhjä, jotta peli toimisi mielekkäämmin, ja pattitilanteen sattuessa kierroksen saa päätettyä ns. skippaustimerin, jonka lisäsin. Skippaustimer päättää kierroksen, kun pelissä on tapahtunut 4 peräkkäistä skippiä, eli neljän pelaajan ollessa pelissä kun jokainen pelaaja on skipannut vuoronsa. Tämän lisäksi jakajasysteemi on muuttunut, eli jakajana toimii kierroksen päättyessä seuraava pelaaja viimeisestä. Pelaajien määrä on pysynyt samana, eli pelin luoja voi päättää uutta peliä tehdessään 2-4:n pelaajan väliltä. Valitettavasti ajan puutteen takia en kerennyt ohjelmoida projetkiin loadgame toimintoa, vaikka se olikin vaatimuksena. Mielestäni projekti on toteutettu helpon- ja keskivaikean välitasona, juurikin pelin latauksen puuttumisen vuoksi.

Käyttöohje

Ohjelma käynnistetään mainista, josta se laukaisee GUI:ssa sijaitsevan mainwindow classin, joka tekee pääasiassa kaiken työn muiden ohjelman apuluokkien kanssa. Ohjelmalla voi pelata nimensä mukaisesti yhtä kasinopeliä 2-4 pelaajalla, ja pelin käynnistäessä voi valita uuden pelin ja pelin latauksen väliltä, joista vain uuden pelin teko toimii. Uutta peliä tehtäessä ohjelman käyttäjä voi valita 2-4 pelaajan väliltä, ja ruudun näkymä muuttuu. Tällöin käyttäjälle aukenee mahdollisuudeksi nostaa kortti, tallentaa peli, tai valita kädessä oleva kortti jonka haluaa pelata. Kortin valitsemisen jälkeen pelaajan täytyy valita pöytäkortti/kortteja, ja valinnan jälkeen painaa turn ready näppäintä, joka tarkistaa liikkeen laillisuuden. Jos liike on laiton, pelaajan vuoro ns. resettaa, ja hän voi jälleen päättää joko nostavansa kortin, tai tekevänsä uuden valinnan. Clear näppäin on koko ohjelman ajan käytössä, ja sillä voi tyhjentää konsoliin tulevan tekstin.

Konsoliin on pyritty tulostamaan ohjeita pelaajalle, esimerkiksi kenen vuoro on, joka havainnollistetaan myös muuttamalla pelikorttien väriä, koska koin tämän itse tarpeelliseksi. Olen lisännyt peliin ns. quality of life tyyppisiä asioita, kuten jos hoveraat hiirtäsi kortin päällä, näet tietoja kortista jota tarvitset pelatessa. Tämän lisäksi ohjelma tulostaa jokaisen kierroksen jälkeen tämänhetkisen pelitilanteen, ja ilmoittaa saamasi pisteet tekemästäsi siirrosta. Pisteissä ei ilmoiteta pelipisteitä, vaan se määrä kortteja jotka lisätään pakkaasi ns. yhtä pelipistettä varten. Pelipisteellä tarkoitetaan kokonaispisteitä, joita pelaajan täytyy kerätä 16.

Ulkoiset kirjastot

Ohjelma ei käytä ulkoisia kirjastoja, poislukien PyQt5. Ohjelma kuitenkin hyödyntää pythonin sisäisiä kirjastoja, kuten randomia pakan arpomiseen sekä functoolsia näppäinten kanssa.

Ohjelman rakenne

Ohjelma voidaan oikeastaan luokitella kolmeen luokkaan.

Pääluokat:

Pääluokilla tarkoitan lähinnä ohjelman tärkeimpiä luokkia, jotka pyörittävät alaluokkia. Näihin kuuluu:

-main.py

Mainin tarkoitus on vain pyörittää graafista liittymää.

-GUI.py

GUI:n sisällä on yksi luokka nimeltä Window, ja se perustuu Qdialog luokkaan. Luokka sisältää n. 400 riviä koodia, ja se koostuu ohjelman piirtämisestä näytölle, sen ylläpitämisestä realiajassa, muutamasta apualgoritmista ohjelman pyörittämiseen ja erilaisista ohjelman päärakenteista, kuten pistelaskurista. Suurin osa funktioista kutsutaan jokaisen vuoron aikana, ja ne ovat riippuvaisia graafisesta liittymästä, jonka takia ne sijoittuvat tähän luokkaan. Ohjelmaan olisi varmasti voinut tehdä useita graafisia apuluokkia, mutta kokemattomuuteni PyQt5:sen kanssa ajoi minut pelaamaan varman päälle, ja sen takia luokka paisuikin näin suureen mittakaavaan.

Alaluokat:

Alaluokilla tarkoitetaan tässä ohjelman pienempien kokonaisuuksien kokoamiseen tiettyihin luokkiin, jotta ohjelma pysyisi selkeämpänä. Näitä ovat esimerkiksi:

-savegame.py

SaveGame luokka hoitaa ohjelman tallentamisen

-gamestate.py

GameState kutsuu newgamea sitä tarvittaessa, ja vastaa esimerkiksi pelimuodon valinnasta, ja palauttaa tarvittavat arvot takaisin GUI:lle.

-newgame.py

NewGame luo uuteen peliin pöydän, pakan ja pelaajat hyödyntäen perustasonluokkia.

Perustason luokat:

Perustason luokat ovat ohjelman ensimmäiseksi tehtyjä luokkia, joiden komentoja hyödynnetään ohjelmassa tarpeen mukaan. Näitä ovat kaikki jäljellejäävät luokat, eli:

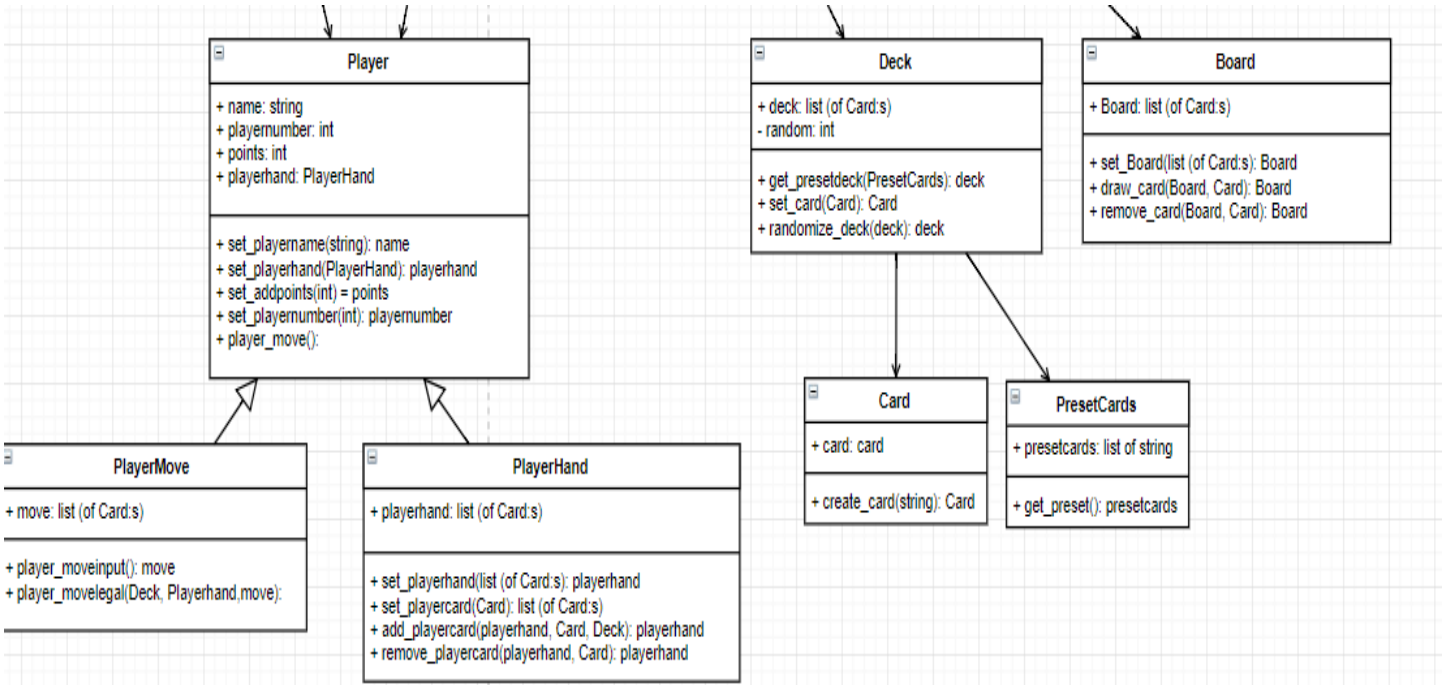
-Board: vastaa pöydän luonnista pakan avulla, ja ylläpitää sitä

-Deck ja PresetCards: luovat pakan ennaltamäärätyistä korteista, ja antavat sen ohjelman käyttöön.

-Player ja PlayerHand: ohjelman pelaajaluokat, jotka jälkeenpäin katsottuna olisi voinut yhdistää, koska siirryttäessä pois tekstipohjaisesta versioista Playerluokalle ei ollut enää juurikaan tarvetta.

-Card: Pelin perustan kannalta tärkein luokka, ottaa presetcardista annettuja stringejä, ja muuttaa ne muille luokille käytettävään muotoon.

Kaikki luokat muodostavat yhteisen kokonaisuuden peliä varten, jossa Deck hyödyntää Card ja PresetCardsia, kun taas Deckkiä hyödyntää Board ja Player/PlayerHand. Näitä luokkia hyödyntää lopulta kaikki muut ohjelman osat. Tein teknistä suunnitelmaa varten UML- luokkakaavion, mutta en näe sitä tässä dokumentaatiossa tarpeelliseksi, vaikka useat luokat ovatkin poistuneet kaaviosta.



Sen sijaan voin käyttää kaavion alaosaa, joka pitää vieläkin enemmän tai vähemmän paikkansa. Playermove on ainut näistä joka ei ole enää käytössä, sekä pelaajia ei enää nimetä, ja pisteet ovat listana joista vastaa playerhand. Tämän lisäksi funktioiden nimiä on muutettu.

Algoritmit:

Funktio hyödyntää useaa algoritmia, joista tähän kerron mielestäni parhaista. Lainaan tähän teknisestä suunnitelmasta ihmisversion tarkistuksen, koska se on pysynyt täysin samana, poislukien tapa jolla inputit annetaan, nämä vaihdan tekstiin.

Ihmisversion tarkistusalgoritmi on suhteellisen yksinkertainen, sillä suurin työ tehdään luodessa jokainen kortti. Ihminen valitsee ensin kortin, jonka haluaa pelata kädestään, jonka jälkeen hän valitsee yksi kerrallaan kortit, jotka hän haluaa ottaa pöydältä. Kyseiset kortit valitaan yksi kerrallaan. Johtuen tavasta jolla kortit annetaan, ne ovat aina olemassa, joista funktio palauttaa listan, jossa ensimmäisenä alkiona on kädestä syötetty kortti, ja toinen alkio on lista, jossa on pöydästä saatavat kortit. Muussa tapauksessa funktio palauttaa arvon False, joka tarkoittaa että prosessi epäonnistui.

Tämän jälkeen kyseinen lista viedään funktiolle, joka tarkistaa kyseisen peliliikkeen laillisuuden. Aluksi funktiossa tarkistetaan löytyvätkö annetut kortit pöydästä ja kädestä, ja jos näin ei ole, funktio palauttaa False ja ilmoittaa virheestä. Jos kuitenkin kortit ovat olemassa, funktio ottaa ensin käsikortista ”käsiarvon”, ja siirtyy laskemaan pöydältä nostettavien korttien määrän. Tämän jälkeen ohjelma lisää total nimiseen muuttujaan pöydästä valittujen korttien ”pöytäarvot”, ja vertaa onko käsiarvo yhtä suuri kuin total. Jos näin on, liike on laillinen, ja toiminto suoritetaan, muussa tapauksessa funktio aiheuttaa jälleen errorin, ja palauttaa Falsen. Käsikorttien ja pöytäkorttien tarkistus ei ole ollut tarpeellinen enää graafisessa, mutta päätin jättää sen osaksi ohjelmaa.

Tämän lisäksi ohjelman GUI osasta löytyy funktiot boardLayout ja handLayout, jotka laskevat kordinaatistosta kortin paikan graafisessa liittymässä, ja niiden dimensiot. Algoritmin olisi voinut tehdä huomattavasti selkeämmäksi, mutta päätin olla muuttamatta sitä, koska se toimii täydellisesti. Molemmat funktiot antavat syötteen apuluokalle, cardlayoutalg, joka tulostaa kortin paikan pöydälle. Kortille tulee graafinen liittymä kutsuttaessa updateGUI:ta, joka päivittää ohjelman kaikki graafisen liittymän osat.

Tietorakenteet:

Ohjelmassa lähes kaikki tieto taltioidaan listoihin, koska mielestäni ne toimivat tarpeeksi hyvin. Ohjelmassa olisi korttien kohdalla voitu käyttää esim. sanakirjoja, mutta mielestäni niistä olisi ollut enemmän haittaa kuin hyötyä.

Tiedostot:

Ohjelma ei hyödynnä mitään tiedostoja, poislukien tapaa jolla se tallentaa tiedon, joka on perus .txt tiedosto, jonka nimi on ennaltamäärätty. Siihen se tallentaa ensin tiedon mitä ollaan tallentamassa, jonka jälkeen tallennetaan pakka ja pöytä. Tämän jälkeen pelaajista tarvittavat tiedot, eli määrä, ja pelaajan käsikortit ja pisteet. Paras tapa testata esim. ohjelman kulkua kierroksien aikana on vaihtaa presetcards.py:n kortteja sellaisiksi, että pöydän saa helposti tyhjäksi, esim lisäämällä 12 r2:sta presetcards listaan. Ohjelma on suunniteltu niin, että korttien määrää voi vaihtaa vapaasti, niin kauan kun niitä ei laita vähempää kuin pöytään ja pelaajille tarvitaan.

Testaus:

Ohjelmaan ei tehty erikseen testiluokkaa, koska se toimii graafisen liittymän kautta. Ohjelmaa testattiin jokaisen uuden funktion jälkeen, ja siinä pyrin korjaamaan jokaisen virhekohdan sellaisen ilmentyessä, tai sellaisen mahdollisuuden ollessa olemassa. En tällä hetkellä ole löytänyt yhtään ongelmaa joka kaataisi projektin.

Ohjelman tunnetut puutteet ja viat:

Ohjelma ei ole täydellinen, johtuen siitä, että tämä oli ennenkaikkea itselleni oppimiskokemus. En ole koskaan rakentanut mitään projektia näin laajassa mittakaavassa, joten työmäärä yllätti itseni vaikka arvioinkin tarvitun ajan suhteellisen lähelle oikeaa. Korjaisin luonnollisesti ensin loadgame funktion ohjelmaan, joka olisi suhteellisen helppo lisätä. Siitä lisää seuraavassa osiossa. Koen, että graafinen liittymä on täysin toimiva, koska sen kanssani pelatessa en ole huomannut ongelmia. Ainut selvä vika jota en ole saanut korjattua, on että joskus uuden kierroksen alkaessa konsoli saattaa printata kahdesti kuka on kierroksen ensimmäinen pelaaja. Tämä johtuu siitä, että funktio kierroksen päättyessä suorittaa itsensä loppuun, ja aloittaa saman uudestaan, jolloin tapahtuu päällekkäisyys.

3 parasta ja 3 heikointa kohtaa:

Hyvänä koen graafisen liittymän, vaikka se ei ehkä ole optimaalisesti ohjelmoitu. Tämän lisäksi koen, että ohjelman perusteet on hyvin rakennettu, ja niiden takana oleva idea on hyvä ja toimiva. Näistä koen parhaimmaksi tavan jolla toteutan ohjelmassa kortit, ja tarkistus algoritmin. Pidän myös suunnitelmiani hyvin rakennettuina ja ne osoittavat ohjelmoinnin ymmärtämisestä.

Heikkoina kohtina ovat ehkä GUI.py tiedoston epäselvä rakenne, ja varsinkin näppäinten kohdalla toistuva koodi. Myös kyseisessä tiedostossa on funktioita, jotka olisi voitu toteuttaa jonnekkin muualle. Ohjelman perusluokista löytyy myös funktioita joita ei ohjelman aikana kutsuta kertaakaan. Kommentointia voisi myös olla enemmän. Myös gamestate ei oikeastaan ollut kovin tarpeellinen, johtuen vaihtoehtojen rajoitteellisuudesta.

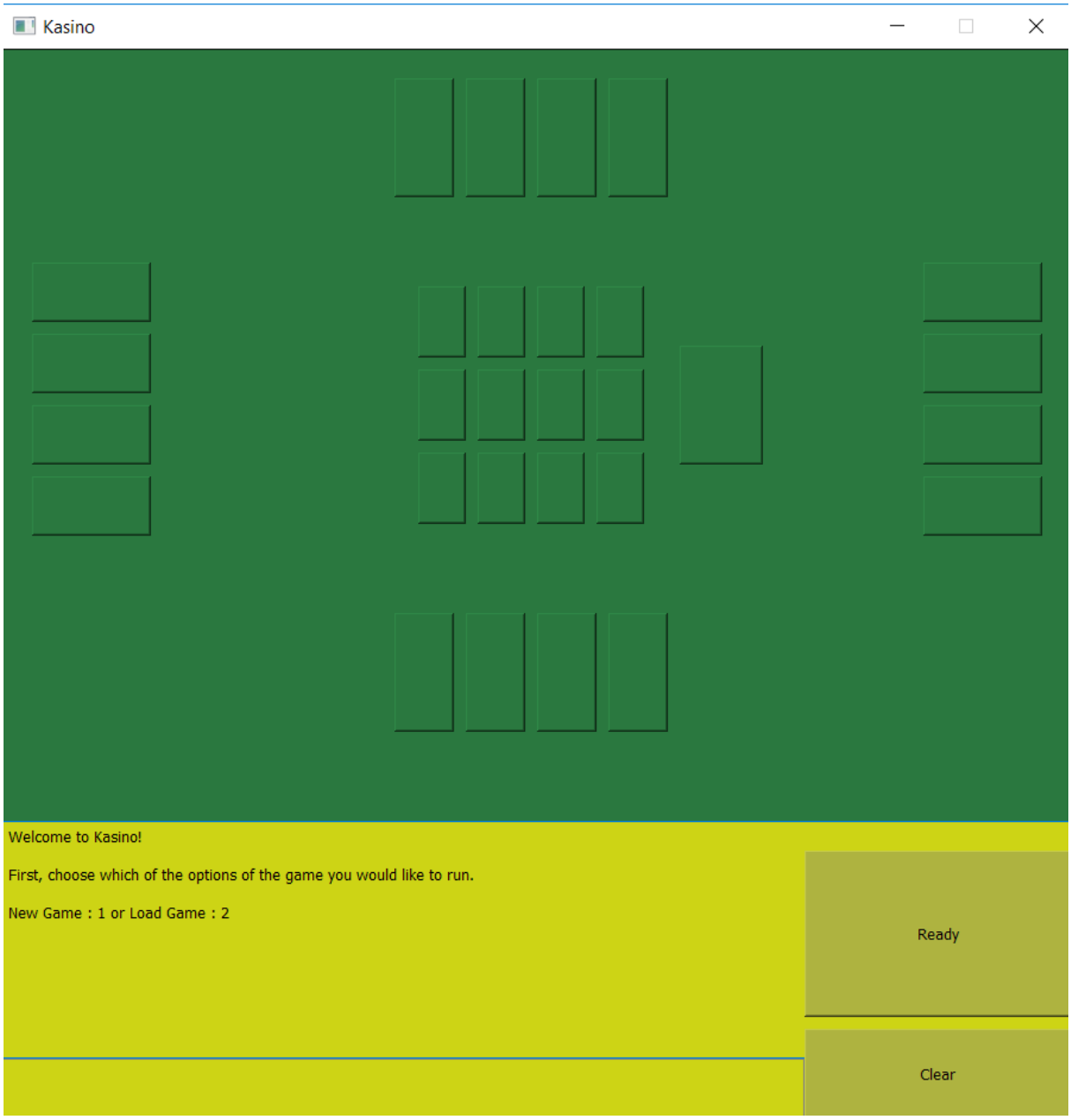
Loadgame featuren puuttuminen on varmasti raskauttava asia, mutta toivon, että arvostelussa otetaan huomioon ohjelman muut hyvät puolet. Loadgamen olisi myös voinut toteuttaa suhteellisen helposti, kun silmäilee millaisessa muodossa data tallennetaan, ja esimerkiksi millä tavalla ohjelma lähtee käyntiin newgamessa. Lataus toimisi, jos savegamen tiedot muutettaisiin listoiksi, ja annettaisiin samalla tavalla createCardsille Window-luokassa, kuin newGame tekee GameStatessa.

Ohjelman virheet olisi voinut korjata ajan kanssa, koska ne olivat kaikki lähinnä optimointivirheitä, poislukien loadgame johon annoin ratkaisun.

Myös yksi heikkous oli tapa jolla nimesin funktiot, koska ne eivät ole yhtenäisiä.

Poikkeamat suunnitelmasta:

Toteutusjärjestys oli mielestäni oikea, ja suunnitelmasta poikettiin ohjelman graafisen liittymän kohdalla, johtuen rajallisesta ymmärryksestäni sen suhteen suunnitelmaa tekiessäni. Myös arvioin tarvitsevani enemmän luokkia kuin todellisuudessa tarvitsin. Ajankäyttö osui oikeaan, koska annoin sen yläkanttiin, kunnes aloin käsitellä graafista osuutta, koska tein n. 3 erilaista versiota siitä, miltä ohjelman pitäisi näyttä. Tässä yksi versioista:



Tässä nähdään versio, missä pelaaja antoi vielä komennot suoraan konsoliin, mutta huomasin tämän olevan suoraan sanottuna surkea idea. Vaikka sain komennot helposti ohjelmasta ulos, niiden käyttäminen järkevästi ohjelman kanssa oli todella vaikeaa. Tämän lisäksi mahdollisten ongelmien huomioon otto olisi koitunut aivan liian isoksi työksi, jonka takia päädyin näppäimiin. Tässä tapauksessa ei tarvitse huolehtia siitä, että pelaaja onnistuisi kaatamaan ohjelman antamalla absurdeja syötteitä. Tämän lisäksi tekoäly ei koskaan poistunut tekstiohjelmasta, johtuen siitä, että en tiennyt miten voisin toteuttaa sen mielekkäästi graafiseen versioon. Teknisessä suunnitelmassa puhuin myös rajattomasta määrästä pelaajien skaalautumisen avulla, mutta en nähnyt sitä tärkeäksi asiaksi ohjelman kannalta, joten päädyin 2-4 pelaajaan.

Toteutunut työjärjestys ja aikataulu:

Ohjelma pysyi aikataulussa ja suunnitelmassa välipalautukseen asti. n. 16.4 alkoi ohjelman graafisen osion suunnittelu, joka muutti suunnitelmaa radikaalisti pois konsolinkäytön luota. Tämän takia yksi ohjelman tärkeistä ominaisuuksista jäi toteuttamatta. Eniten töitä tehtiin 17,18 ja 19.4, jolloin koodasin ohjelmaa päivässä jopa 12 h. Ensimmäiset 2 päivää meni asioita opetellessa, ja viimeisin päivä ohjelman korjailemisessa järkeväksi. Ohjelma toteutettiin siis ensin tekemällä perusta, jonka jälkeen graafinen liittymä.

Arvio lopputuloksesta:

Ohjelma on toimiva kokonaisuus, jonka hyvät puolet ovat graafinen liittymä ja vahva perusta, sekä mielestäni virhekomentojen pieni mahdollisuus. Työn oleellisin puute on latauksen puuttuminen, joka johtui graafisen liittymän haasteista. Tulevaisuudessa ohjelmaa voisi parantaa korjailemalla funktiota tiiviimmiksi, ja yhdistelemällä niitä, koska PyQt5:sen jotkin komennot varmasti olisivat tehokkaampia kuin ne joita valitsin käyttää. Myös näppäinten määrittely oltaisiin voitu hoitaa yhdessä funktiossa, ja graafinen liittymä jakaa useampiin osiin. Jotkin algoritmit olisi voinut tehdä selkeämmiksi, jolloin niitä olisi voinut käyttää tehokkaammin. Ohjelman rakenne soveltuu muutosten tekemiseen, jos sitä tiivistäisi. Tällä hetkellä jos joku alkaisi muuttaa ohjelmaa, voisi herätä kysymys mistä aloittaa? Tämä johtuu mahdollisesti funktioiden epäselvästä nimeämisestä, sekä graafisen liittymän sekavasta rakenteesta. Olen kuitenkin ylpeä siitä, mitä sain aikaan, ja käytin tähän uskomattoman määrän aikaa, johtuen siitä, että halusin oppia. Koenkin, että projekti näyttää enemmän mitä opin tämän matkan aikana, ja jos nyt saisin samanlaisen homman eteeni, omaisin huomattavasti paremmat valmiudet lähestyä ongelmaa realistisesti ja itsevarmasti.

Viiteet:

https://wiki.python.org/moin/

https://wiki.python.org/moin/PyQt/Tutorials

https://docs.python.org/3/library/random.html

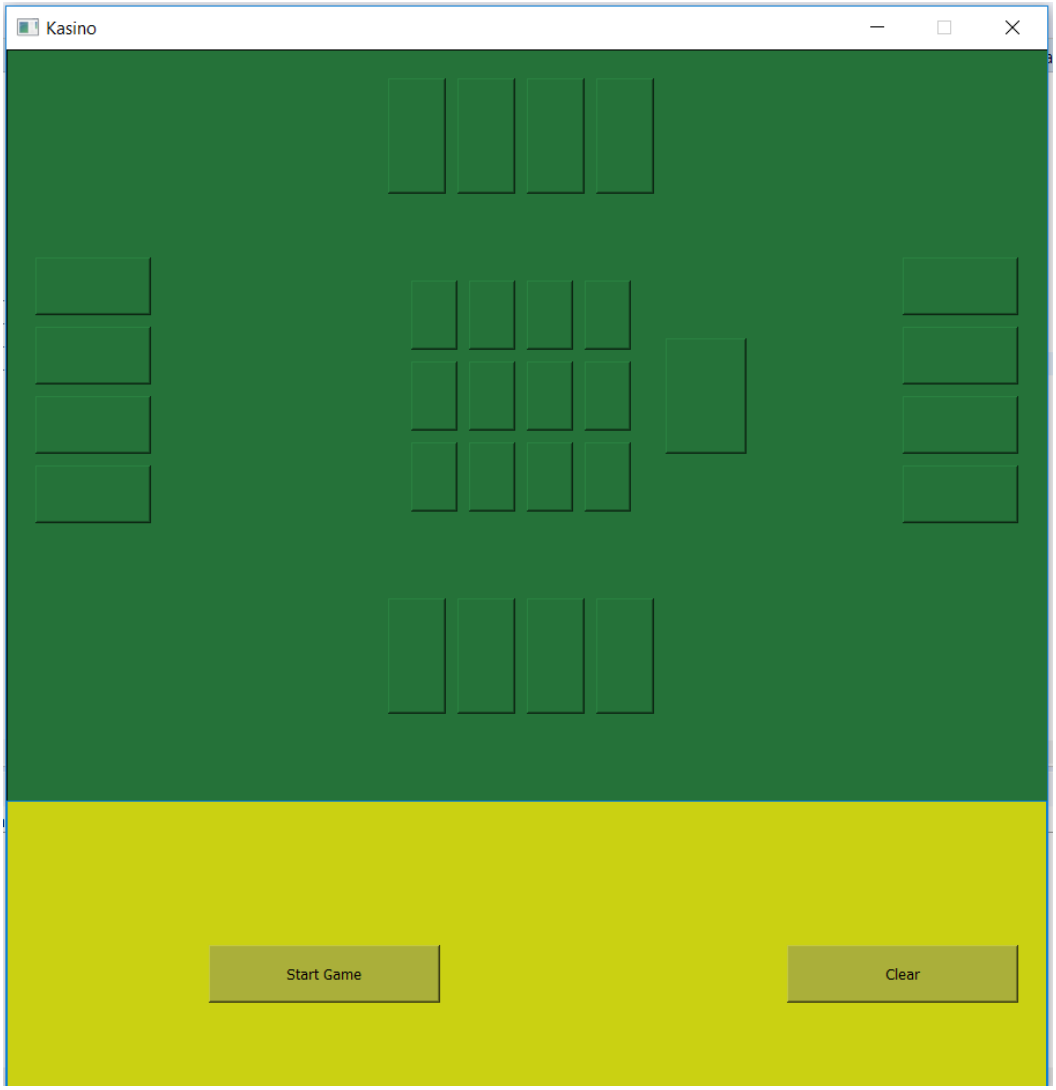
https://www.qt.io/

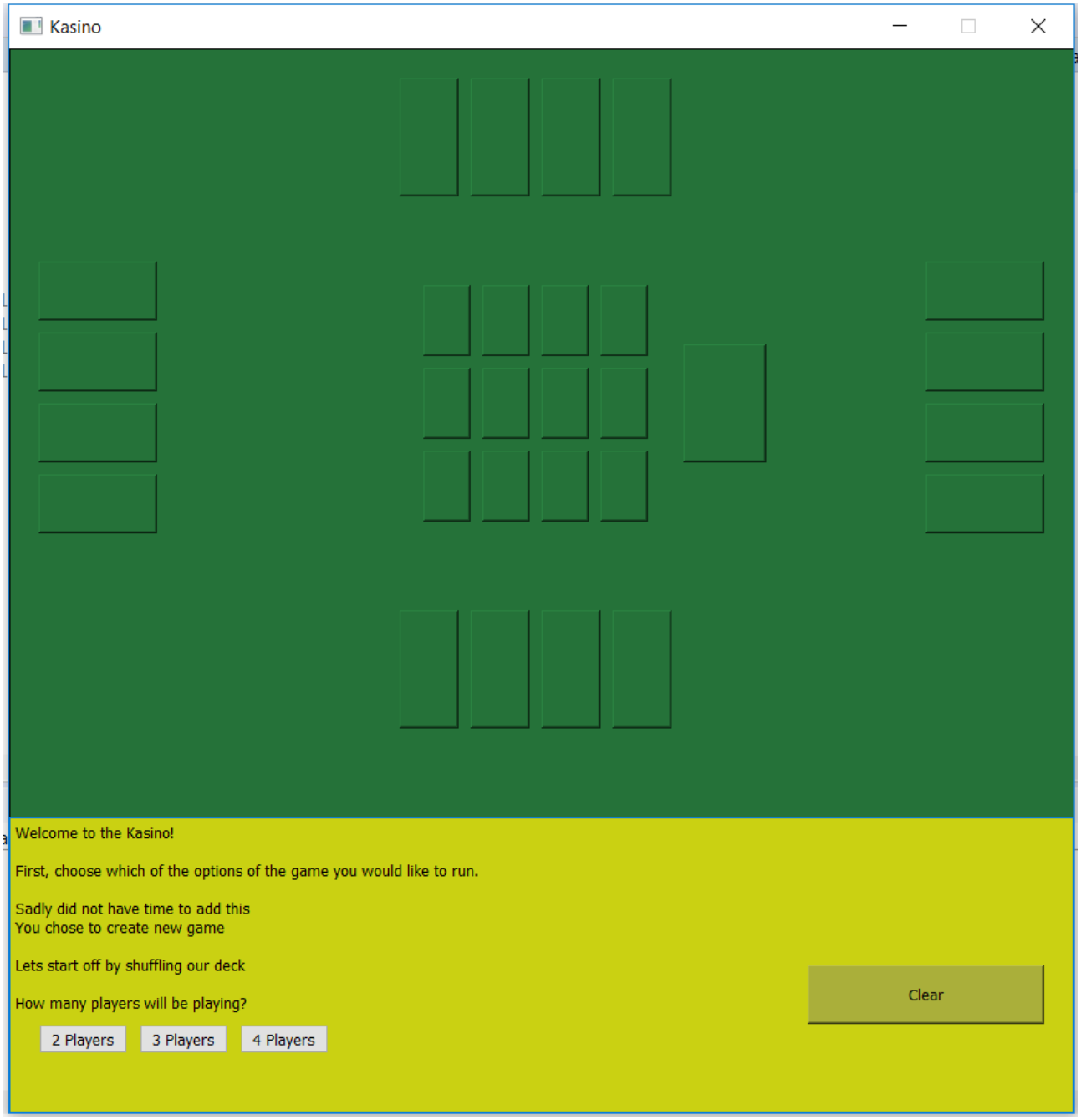
https://stackoverflow.com/

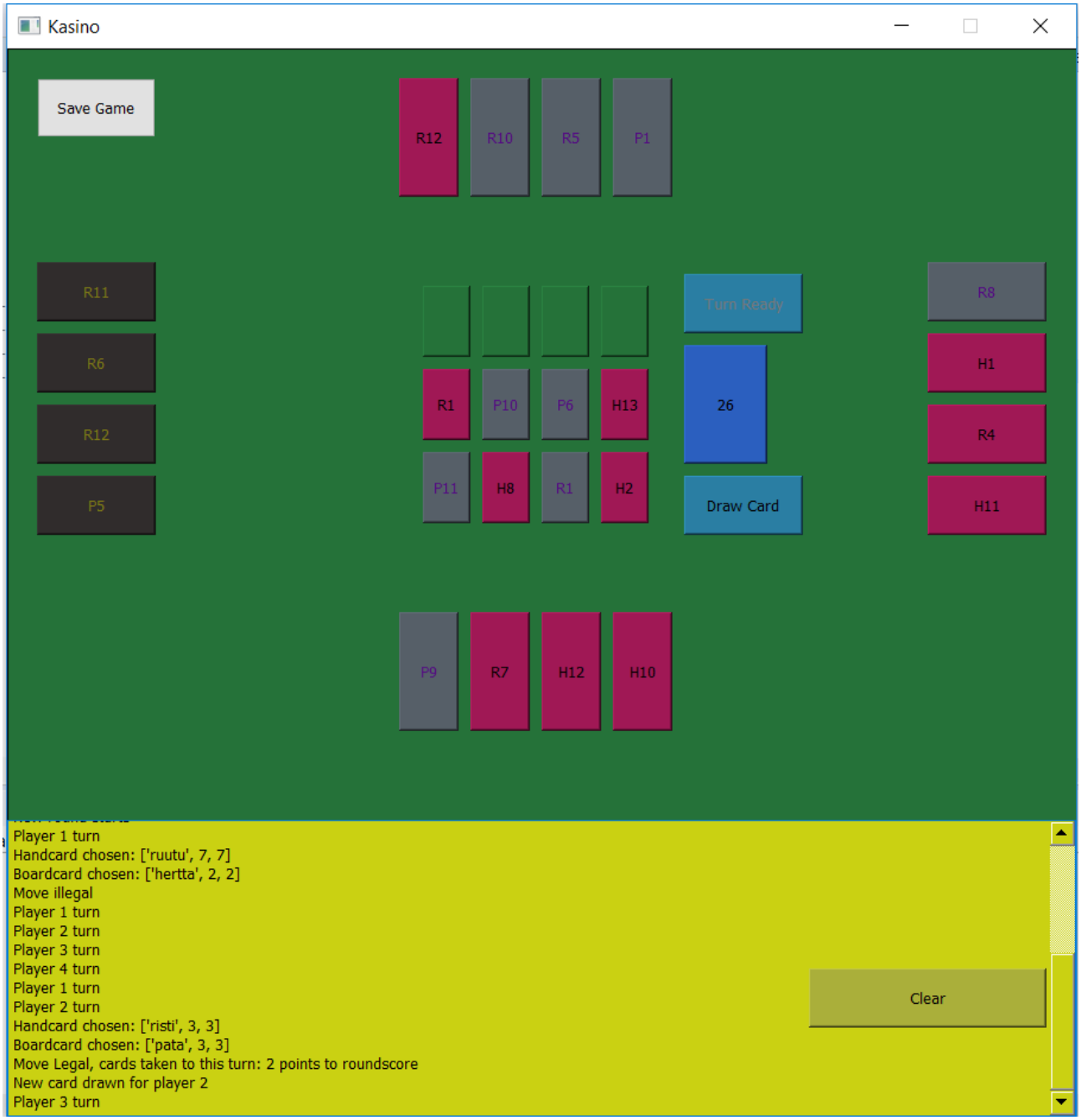
https://www.youtube.com/watch?v=pnpL9Sl79g8&list=PL1FgJUcJJ03uwFW8ys2ov2dffKs3ieGYk

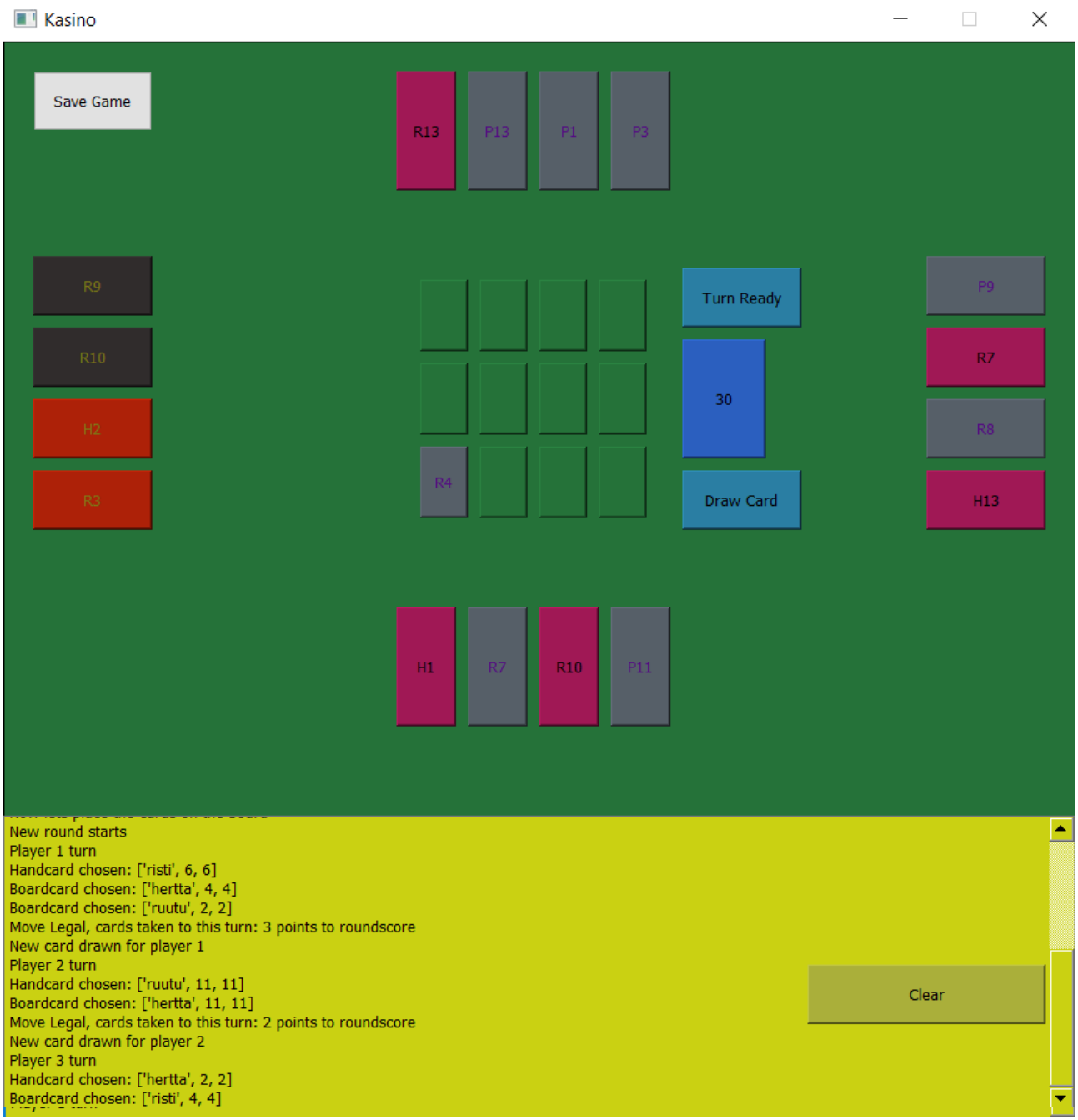
RobotWorld kurssin esimerkkitehtävistä

Liiteet;




Kasino
Welcome to the Kasino!
First, choose which of the options of the game you would like to run.
Sadly did not have time to add this
You chose to create new game
Lets start off by shuffling our deck
How many players will be playing?
2 Players
3 Players
4 Players
Clear


Kasino
Save Game
R12
R10
R5
P1
R11
R6
R12
P5
R1
P10
P6
H13
P11
H8
R1
H2
Turn Ready
26
Draw Card
R8
H1
R4
H11
P9
R7
H12
H10
Player 1 turn
Handcard chosen: ['ruutu', 7, 7]
Boardcard chosen: ['hertta', 2, 2]
Move illegal
Player 1 turn
Player 2 turn
Player 3 turn
Player 4 turn
Player 1 turn
Player 2 turn
Handcard chosen: ['risti', 3, 3]
Boardcard chosen: ['pata', 3, 3]
Move Legal, cards taken to this turn: 2 points to roundscore
New card drawn for player 2
Player 3 turn
Clear


Kasino
Save Game
R13
P13
P1
P3
R9
R10
H2
R3
R4
Turn Ready
30
Draw Card
P9
R7
R8
H13
H1
R7
R10
P11
New round starts
Player 1 turn
Handcard chosen: ['risti', 6, 6]
Boardcard chosen: ['hertta', 4, 4]
Boardcard chosen: ['ruutu', 2, 2]
Move Legal, cards taken to this turn: 3 points to roundscore
New card drawn for player 1
Player 2 turn
Handcard chosen: ['ruutu', 11, 11]
Boardcard chosen: ['hertta', 11, 11]
Move Legal, cards taken to this turn: 2 points to roundscore
New card drawn for player 2
Player 3 turn
Handcard chosen: ['hertta', 2, 2]
Boardcard chosen: ['risti', 4, 4]
Clear