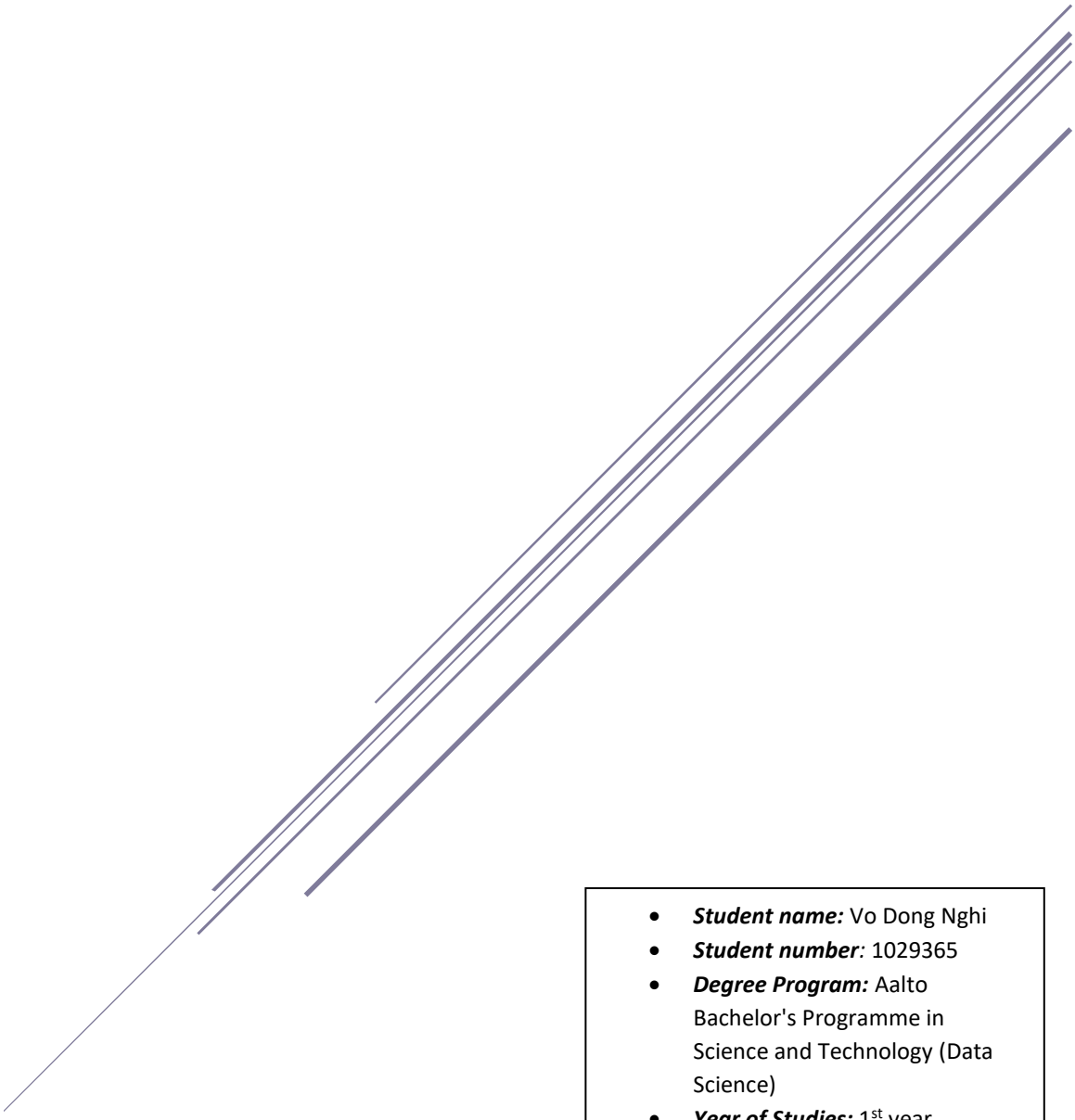
PROJECT GENERAL PLAN

Project Title: Casino



- ***Student name:*** Vo Dong Nghi
- ***Student number****:* 1029365
- ***Degree Program:*** Aalto Bachelor's Programme in Science and Technology (Data Science)
- ***Year of Studies:*** 1st year
- ***Date:*** 13/02/2022

GENERAL DESCRIPTION

The aim of this project is to create a Cassino game, which leads to two essential goals: first, this project should be able to recreate a real-life Cassino game, with all its dealing, playing rules and scoring; and second, this project should take the gaming aspects into account, such as graphical interface, computer-simulated players, saving data of previous game state, …

As for the first goal, this project would comply with the rules of Finish / Nordic Cassino as described in the Cassino project page of A plus and Pagat.com [1]. Specifically, the gameplay could be described in parts, including:

1. Deal: The dealer is picked randomly at the start of the game, which can be one of the players or a computer-controlled player. The cards are then dealt in pairs, one pair to each player, one on the table (face-up) and one to the dealer, then repeat. It is important that the dealing of the cards in the game is the same as that in real-life. Then for each round, the player to the left (clockwise) becomes the dealer. The game starts with the player to the left of the dealer making the first move, then play turn passes clockwise
2. Play: After the dealing, each player has 4 cards (not visible to other players) and 4 cards on the table (visible to all players). Each turn, the player always plays one card to the table. If the player captures a card or cards, the player put the card played and the card (s) captured in a separate pile. If no card is captured, the player plays his or her card on the table face up to be captured later. A played card can capture:
    a. A card on the table of the same capture value, or
    b. A set of cards on the table whose capture values add up to the capture value of the played card, or
    c. Several cards or sets of cards that satisfy conditions 1 and/or 2 above

    If some player gets all the cards from the table at the same time, he or she gets a so called sweep which is written down. There are a couple of cards that are more valuable in the hand than in the table:

    - Aces: 14 in hand, 1 on table
    - Diamonds-10: 16 in hand, 10 on table
    - Spades-2: 15 in hand, 2 on table

    The player must also take a card from the deck so that he or she always has 4 cards.
3. Scoring and End: When every player runs out of cards, the last player to take cards from the table gets the rest of the cards from the table. After this the points are calculated and added to the existing scores. The following things grant points:
    - Every sweep grants 1 point.
    - Every Ace grants 1 point.
    - The player with most cards gets 1 point.
    - The player with most spades gets 2 points.
    - The player with Diamonds-10 gets 2 points.
    - The player with Spades-2 gets 1 point

    One must collect points which are calculated after every round. The game continues until someone reaches 16 points.

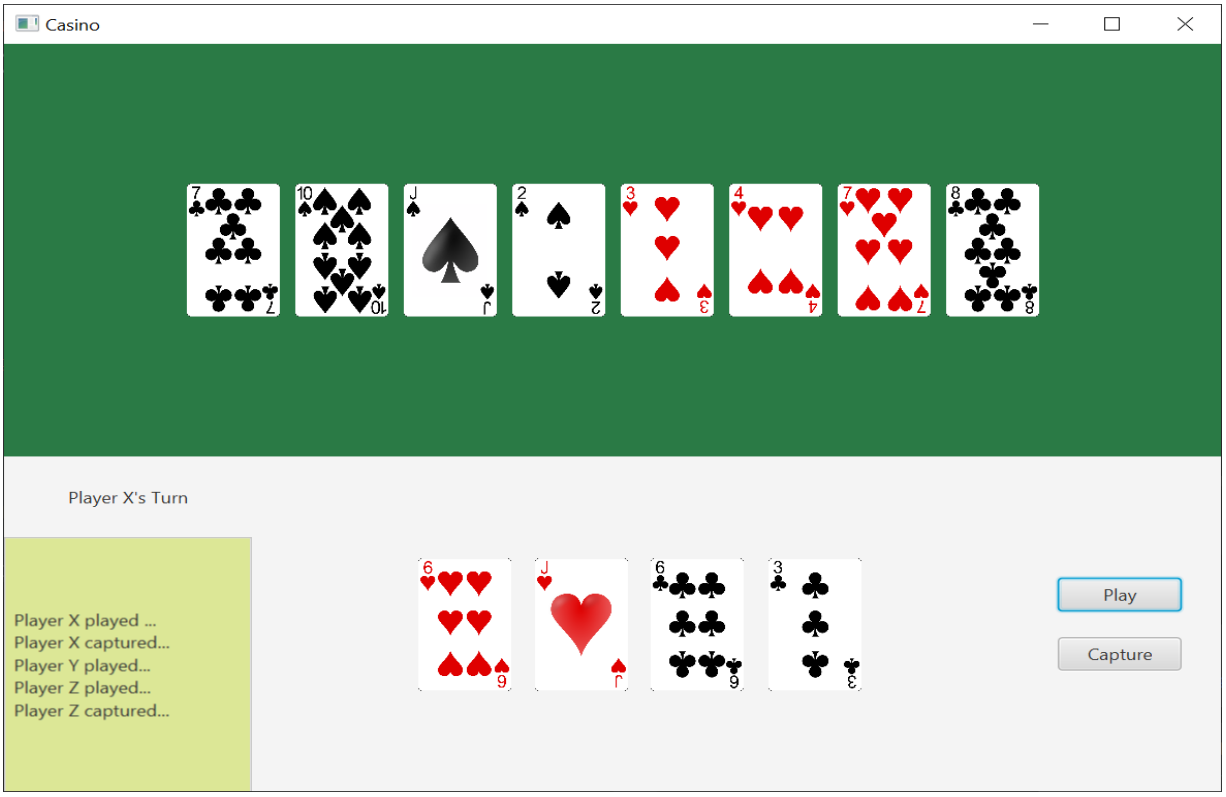
As for the second goal, this project aims to create a game to satisfy the demanding level of difficulty. This game should be implemented with graphical user interface. It is possible to play against human players (2 – N) players and play against computer-controlled opponents (0 – N). The computer-controlled opponents will have access to different strategies, including find sweeps, avoid easy sweeps, save cards for later use, … which they choose with priorities or randomness. The program also should be able to load and save the game in progress, which would be discussed further in the following section.

USER INTERFACE

From the perspective of the player, the necessary information to play the game is the cards on his or her hand and the cards on the table, including what cards are currently on the table, what cards were captured by what cards and what players, and the resulting score after each round. As for the program, it needs the information about what move the player makes in his or her turn, including what card is played and what card (s) is captured. These aspects will be taken into account with regards to the output and input. The output will include the necessary information for the players in form of pop-up messages, text and images, such as pop-up message about the winner, scores of each round, text about the turn, images of the cards on the hand and cards on the table. Meanwhile, the input is obtained with mostly buttons and clicks, such as clicks on a card will add that card to a list/buffer/sequence of chosen cards, buttons for actions like play a card, capture, …

Apart from the game play, there are still additional input to be obtained at the start or end of the game. For example, at the start, the player can choose either to continue from the previous game or start a new game. The player can also choose to play with human players or with computer-controlled player. When the game ends, the player can choose to start a new game or quit. This type of input can be obtained through buttons from opening menu for instance, or quit button

The game screen will focus mostly on the current player's hand and the visible cards on the table. There will be buttons for capture and play (plays a card without capturing). The player chooses one card on his or her hand and the button 'play' to play the card without capturing. The player chooses one card on his or her hand and a card or cards on the table and the button 'capture' to capture the cards. There will be an indicator for the player's turn and which player is having his or her turn. There will be an event log to summarize the turns of each player. There would also be a message to announce the current scores after a round.



This is a draft user interface view for the player during a round.

FILES AND FILE FORMATS

The file loading can be divided into loading images and loading current game information, of which the latter is essential. The current game information will be stored as a txt file, in a format similar to that in the ChunkIO or HumanWritableIO exercise. The data stored will contain game mode (vs human players or computer-controlled players), the number of players, name (s) and type of players, scores and the cards, including cards at each player's hand and captured pile, cards on the table, cards remaining in the untouched pile. In this way, the data file can be stored as a simple txt file and can be easily read using FileReader. One format the file can be stored in is:

#game metadata

mode: human/computer

numPlayer: 2-N /*a number greater or equal to 2*/

cardOnTable: sasjh1d8…

/*Each card is described in 2 characters, the first is the initial letter of its suit (Club – C, Diamond – D, Heart – H, Spades – S) and the second is the initial letter for Ace, Jack, King, Queen or the value of the card from 2 to 9 with the exception of 10 being shortened to 1.*/

cardInPile: … /*similar to above*/

currentTurn: 1 - N /* player's order of creation

currentDealer: …

currentLastCapture: …

#player1

order: 1 - N /*The order of creation, does not involve with the game play (not the order of turns), used to differentiate between players

type: human/computer

name: [String] /*computer-controlled players have random names*/

score: 0-16 (or more if tied)

sweep: 0-N /*numer of sweeps

cardOnHand: …

cardInPile: …

#player2 …..

/*The game metadata still contains the information about the number of players to check if there are missing players.*/

As for the loading images, this task mostly involves loading the cards. One available source of images is OpenGameArt.org [2] with all necessary deck images. These images are all PNG files and can be loaded using Image and ImageView of ScalaFX.

REFERENCE LIST

[1]: https://www.pagat.com/fishing/nordic_casino.html

[2]: https://opengameart.org/content/playing-cards-vector-png