**Project 2**

**<Tycoon>**

**CSC-17C**

**Name: Choy, Arthur**

**Date: 6/4/21**

**Introduction**

For my project, I coded the game “Tycoon,” otherwise known as “Daifugō,” in Japan. As implied, it is a card game that originates from Japan. I originally chose this game as I stumbled upon it while browsing on YouTube, in which a variant of the game was featured within another game known as Persona 5: The Royal. I decided it looked interesting enough to try to program.

Development time: about a week

Lines: 1358

Classes: 4

Structures: 1

Significant functions: 50

Significant variables: 20

GitHub Link: <https://github.com/ac2860356/2020_Fall_CIS_CSC_17C/tree/master/Project/CSC%2017C%20-%20Project%202>

**Approach to Development**

The most difficult part was trying to find places to implement the concepts in the program. I couldn’t fathom a way I could add graphs (that weren’t trees), for example.

For versioning, I just decided to make a new version whenever I made significant changes to my program over a period of time, which in this case meant whenever I changed the program to support different concepts

Version Breakdown:

Version 1 – Recursions/Recursive sorts

Version 2 – Trees/Hashes

**Game Rules**

(Disclaimer: some of the rules for the game in the program are different from the rules that you’ll find online because I misread them and by the time I realized, it was too late to change. I will be stating the rules that I programmed). Tycoon is a card game in which the objective is to be the fastest player to play your entire hand. The game consists of 3-5 players.

Titles (Explanation, and mechanical purpose):

Players in Tycoon are able to get titles for each round. A player’s title is determined by how quickly they clear their hand. There are five titles: Millionaire, Noble, Commoner, Poor, and Destitute. At the beginning of the game, every player will start as a commoner. The fastest player to clear their hand becomes the Millionaire, the second the Noble, etc. When there are four players present, no player will be a Commoner after a round; when there are three players present, no one will be a Noble or Poor. The players with the richer titles will have advantages over the players with the poorer titles when a round begins. Though those advantages/disadvantages change depending on rules set by the players, in the program, the Millionaire is allowed to go first in a round. Also, before every round, after cards are dealt, the Millionaire is allowed to get rid of their worst two cards and give them to the Destitute, who has to give their best two cards in return. The Noble and the Poor do a similar process, except with one card. This card trade is known as “taxing”.

Playing:

In Tycoon, the face value of cards are shifted such that 2 is the highest ranking card, and 3 is the lowest. Every round in a game Is broken up into tricks. Every trick is started when the player starting the trick places down any amount of cards of the same rank that they have in their deck. Subsequent players must play the same amount of cards with a higher face value (for example, if the trick is started with a pair of 3’s, the next player must play a pair of 4’s or higher). A trick ends when all except one player has passed, or all except one player is left playing. If everyone except one player has passed, then the player that didn’t pass will begin a new trick. The round ends when all except one player has cleared their hand, in which titles will be handed out as stated earlier. Also, players are allowed to pass on their turn without direct penalty.

Winning:

The winning player is usually who ends up as the Millionaire at the end of the game. Players are allowed to play as many rounds as wanted in order to determine this.

**Description of Code**

My code is broken into three major classes, which constitute most of the program and function of the game: The Deck, the Player, and the Game.

The Deck contains the standard cards in a card deck (without Jokers, and suits are not considered), which is randomly shuffled. The card at the top of the deck can be pulled for card dealing, as would be done in real life.

The Player contains individual data that would pertain to the player, such as their name, the cards in their hand, and their title. The class also has internal functions to play cards in a hand, and also getting the best or worst cards in a hand for taxing.

The Game class is the crux of the game, as the name would entail. It contains a list containing all the players. There is a hierarchy of functions; the game function executes all of the rounds in a game and deals with the titles and taxing after a round. Rounds are consisted of repeated trick functions, which execute tricks in the manner described in the game rules. Each trick consists of repeated card playing functions, which contain input validation while a player attempts to play cards.

**Sample Input/Output (only contains a snippet as to play the whole game through would take too long)**

*Output:*

1. Start a Game

2. See How to Play

3. Test Game Functions

4. Quit Program

Please select an option by typing in a corresponding number:

*Input:* 1

*O:* How many players are playing? (3-5 players only):

*I:* 5

*O:* Please enter the name for player 1:

*I:* qwe

*O:* Please enter the name for player 2:

*I:* wer

*O:* Please enter the name for player 3:

*I:* ert

*O:* Please enter the name for player 4:

*I:* rty

*O:* Please enter the name for player 5:

*I:* tyu

*O:* Please enter the amount of rounds to be played:

*I:* 3

*O:* Please enter the cards in a hand (max 10 per player, min 3):

*I:* 6

*O:* A new trick begins

qwe's Turn

9 3 K K A Q

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 3

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: 3

*O:* wer's Turn

J 6 10 Q J 4

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 4

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: 4

*O:* ert's Turn

J A 5 2 8 8

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 6

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* You don't have that card in your deck

*O:* Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 5

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: 5

*O:* rty's Turn

4 9 9 K 4 6

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 6

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 2

*O:* Must play the same number of cards that was previously played

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* y

*O:* That is not a valid rank

*O:* Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 6

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: 6

*O:* tyu's Turn

A Q 5 5 8 J

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* qwe's Turn

9 K K A Q

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* A

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: A

*O:* wer's Turn

J 6 10 Q J

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* ert's Turn

J A 2 8 8

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* 2

*O:* Please enter the amount of cards of this rank you want to play (inputs higher than the amount you have will play all of them):

*I:* 1

*O:* Played cards: 2

*O:* rty's Turn

4 9 9 K 4

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* tyu's Turn

A Q 5 5 8 J

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* qwe's Turn

9 K K Q

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* wer's Turn

J 6 10 Q J

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

*I:* Pass

*O:* A new trick begins

ert's Turn

J A 8 8

Please enter the rank of the card you want to play (if you want to pass, type Pass; if you want to sort your hand, type Sort):

**Checkoff Sheet**

|  |  |
| --- | --- |
| **Concept** | **Location** |
| Recursions | Game.cpp : line 164 |
| Recursive Sorts | Game.cpp : line 269 |
| Hashing | AVLTree.cpp : line 25 |
| Trees | AVLTree.h & AVLTree.cpp |
| Graphs |  |

**Documentation of Code**

Flowchart and UML Class Diagram included in Project Folder

Pseudo-Code

***Classes***

*Deck: contains 4 cards of every face value, can be drawn from like a regular card deck and shuffled*

*Player: contains player hand, name, and title value*

*Game: contains core game functions, along with the list of players and a deck*

***Functions***

*draw(), draw a card from the top of the deck*

*insert(), place a card into the deck*

*addHand(), add a card to the player’s hand*

*clearHand(), clear the player’s hand*

*empt(), return whether or not the player’s hand is empty*

*playCard(), have the player play a card or a group of cards of the same face value*

*getBest(), get the best cards in a player’s deck*

*getWorst(), get the worst cards in a player’s deck*

*start(), core function of the game; will play out all rounds and will handle titles*

*trick(), begin a trick*

*playCards(), let a player play cards*

*dealCards(), deal cards to the players*

*cardTrade(), trade cards based on rank*

***Initialize game***

*Get player amount (limit to 3-5)*

*Get player names*

***Begin a round | start()***

*Deal cards to players | dealCards()🡨 addHand() 🡨 draw()*

*trick()*

*First player that plays cards gets to play any amount of the same face value, as long as they have it in their hand | playCards() 🡨 playCard()*

*Next player must play same amount of cards of higher face value | playCards() 🡨 playCard()*

*When a player finishes their hand, give them the highest unrewarded title | empt()*

*If all except one player passes, begin a new trick | trick()*

*When all players except one have finished their hand, end the round, give the last player the lowest title*

*Clear player hands*

*Initiate card trade | cardTrade() 🡨 getBest()/getWorst()*

*If there are 3 players: trade 2 cards between the highest and lowest player*

*If there are 4 players: trade 2 cards between the highest and lowest player, and 1 card between the remaining two players*

*If there are 5 players: trade 2 cards between the highest and the lowest player, and 1 card between the second highest and second lowest*

*The next round will begin with the highest player*

*Repeat for as many rounds as the player wants*