Sheridan College

High vs Low Card Game: Project Design

SYST 17796: Fundamentals of Software Design and Development

Thomas Wong, 991623854

March 21, 2021

Table of Contents

Project Background and Description

This project aims demonstrates the member's ability to create/extend new coding ideas based on a set of given starting code. By following object-oriented programming principles, the members are to build a simple card game that extends from the *Project Starter Code* presented in this course.

The game this project aims to implement is a standard 52-card game between 2 players where:

- Each player will have a starting hand size of *5* unique cards from the deck of 52 cards,
- A dealer will play 1 card from the remaining deck of 52 cards.
- Each player must then deal a/any card from their hand.
- A comparison will be made between the player dealt cards and the dealer card:
    - The player with the highest value card that wins the round
- Both players draw 1 from remaining card deck
- The round will repeat until dealer plays the last card from deck
- The player with the most points will win the game.

The card values will be determined by this formula:

$$\boldsymbol{Card\ Value} = Suit + Card\ number$$

$$Where, Clubs = 0, Diamonds = 13, Hearts = 26, Spade = 39$$

$$Ace = 1, Jack = 11, Queen = 12, King = 13$$

The starting base code of the card game is written in java.

Project Scope

The team leader will be implementing and extending existing code to meet the needs of the above tasks specified. This project will be created by extending(inheriting) the base starter card with more defined subclasses. Eg. The StandardCardDeck class will model the set of standard 52 card deck which inherits from the Card class. The Card class models all sets of playing card. The project will be completed when all the below High-Level Requirements are satisfied.

High-Level Requirements

The implemented system must include the following:

- Ability for each player to register their name with the game.
- Ability for the game track player score.
- Ability for the players to track cards dealt on each round.
- Ability for the players to deal from their own hands.
- Ability for the game to present the winner of a round and game.

Implementation Plan

The project will be stored and updated using a Git repository. The repository will contain all the deliverable files related to this project. The team leader will be using Netbeans IDE 8.2 and Git version control system.

Git URL: https://github.com/wongtho/Syst17796_Del1.git

This project will be implemented using the Random class to simulate user decisions. The playerID will be hardcoded for easier testing throughout the development process. Thus, error checking will not be implemented until user inputs are integrated into the code. All random class variables and playerIDs can be replace by the Scanner class for user input later.

Design Considerations

The Starter code provided in this project is a skeleton model of a card game with no specific method that can used to differentiate itself from any given card game. The minimum required are the playerNames and gameName. Each new class specific to this project game will be implemented by inheriting its respective parent class. Eg. HighvsLowGame class will inherit from Game class, PlayerHighLow class will inherit from Player class. Encapsulation will be used to protect variables used within its class. Getter & Setter methods will be used to access these protected variables.

- The GroupOfCards class will be used to model a hand/set of cards at any given moment. Thus, three additional methods will me implemented:
    1. Deal method to simulate choosing/submitting a card from hand.
    2. Draw method will allow the hand/set of cards to receive a card.
    3. FullDeck method to generate all possible cards into hand.
- StandardCardDeck models any card in a 52 card deck. This class will inherit from the Card class and is refactored from Card class in ICE 1 and 2.
- Player class will include a score variable to track player score
- HighvsLowGame class will inherit from Game class. This class specifies the functionality of the game. Additional methods will me implemented in this class to improve readability, code cohesion and coupling.