Java Android Project Manual

Organization of Programming Languages

Professor Kumar

04/25/2020

Salil Maharjan

# 1 Bug Report

The program does not have any known bugs.

# 2 Feature Report

## 2.1 Missing Features

All features reported on the project description and grading rubric from the website has been successfully implemented.

## 2.2 Extra Features

The user player has the option to use a custom name in the game. If no name is entered, the player name defaults to “User”.

The capture pile is sorted as mentioned in the grading rubric and it is extended so that the hand pile is also sorted for ease of readability.

Stacks and recommended cards are highlighted elegantly.

The computer strategy and user help mode, considers the stacks on the player’s capture pile to give recommendation for the next move.

# 3 Data structure/Classes Description

There are seven classes used in the program: Game, Round, Player, User, Computer, Card, Deck.

Game is the main class to start the game. It creates Round class objects as requested by the user and stores information like the round number, total player scores, and the Player class objects. It includes a method to load a saved game from a configuration file.

The Round class holds all necessary methods to start a round of the game. It stores information about the cards on the deck and the layout during each round. It also includes a method to save file which can be used on any round to save the game to a configuration file. As Game and Round class are highly coupled, the Round class is made a friend of Game class so that it can access and update member variables like total score for each player and the round counter. This also gives access for the Round class to access the Player objects who are playing in the game. The Round class also uses the Deck class for creating a stock pile and to distribute cards.

Player is an abstract class and has virtual methods. User and Computer inherit Player and provide separate implementations of virtual methods declared in Player class. Methods that are common for both User and Computer player are defined in the Player class like sorting and arranging cards on hand and in the capture pile. Separate implementations for other methods which differ for the two players are defined in their respective classes.

The Deck class uses the Card class to create a standard deck of 52 cards. The Card class holds information about the card face and suit.

# 4 Log

January 22, 2020:

* Listed out main classes to use for go stop game. (30 min)
* Read the description and rules of the game. (30 min)
* Made a design for the C++ project (30 min)
* Wrote and tested Card and Deck class (1 hour)
* Wrote function to shuffle deck and incorporated feature to create Deck objects with multiple deck of cards. (30 min)
* Tested functions and the new deck class. (1 hour)

January 23, 2020:

* Wrote skeleton for classes according to yesterday’s design with member variables. (1 hour)
* Wrote constructors Game, Round, Player- User and computer classes (1 hour)
* Implemented the beginning of the game to distribute cards to players (1 hour)
* Removed inheritance of Round class to Game because Game constructor was being called every time Round is constructed. (1 hour) Tried making them friends but finally replaced it by passing the required variables through the constructor (1 hour)
* Tested card dealing function. (1 hour)
* Replaced random shuffle function with shuffle function because it is deprecated in c++11 (30 min)

January 24, 2020:

* Had to pass too many parameters between Game-Round classes. Redeclared Round as friends of Game to avoid parameter overloading. (30 min)
* Wrote function to determine first player according to the game rule. (1 hour)
* Wrote the main game loop for the rounds. (1 hour)
* Made Round and Game constructors stronger by initializing required variables for the game. (30 min)

January 25, 2020:

* Wrote function to deal cards. (1 hour)
* Changed vector for layout into vector of vectors as cards can be stacked. (1 hour)
* Completed dealing to players and layout function. (1 hour)

January 26, 2020:

* Wrote function to display board in the command line (1 hour).
* Wrote function to process the turns in the round according to the determined first player (1 hour)
* Wrote overloaded play function for user. Finished the card picking code. (1 hour)
* Wrote function to check picked card with matching cards in the layout. (1 hour)

January 27, 2020:

* Wrote function to complete card selection and processing it on the layout. (1 hour)
* Debugging to find and process triple stacks: H3 (1 hour)

**Total: 20.5 hours**

January 29, 2020:

* Complete function to process user hand moves. Cases h1-h3. (1 hour)
* Completed function to deal cards from stock pile. (1 hour)
* Completed function to handle cases after drawing from stock pile (1 hour)
* Added input validation (30 min)

February 1, 2020:

* Completed play function for computer player. (1 hour)
* Completed strategy function and capture function. (1 hour)
* Introduced game score and round score. (30 min)
* Worked on the GUI (30 min)
* Worked on helper option for user (30 min)

February 2, 2020:

* Completed function to save game (30 min)
* Completed function to read from serialization file and load game. (1 hour)
* Formatting serialization file and debugging (30 min)

**Total: 9 hours**

February 5, 2020:

* Added feature to display selected card and layout matches. (30 min)
* Refactoring code. (30 min)
* Rewrote score loader to save score as total and recalculate round score. Debugging and completion. (30 min)

February 10, 2020:

* Checking grading rubric and refactoring code. (30 min)
* Displaying board in between hand move and stock pile move (30 min)
* Wrote function to sort player’s capture pile and tested it (30 min)
* Code refactoring and put unused methods as “Trash methods” (30 min)
* Rechecked documentation (1 hour)

February 11, 2020:

* Rechecking program and some test cases. (1 hour)
* Writing manual for demo. (1 hour)

February, 12, 2020:

* Made input validation stronger. (30 min)
* Function to process configuration files without hyphens for stacks to behave as usual. (30 min)
* Function to sort hand pile. (30 min)
* Debug and testing (30 min)
* Fixed bug of round ending early without both player’s hand being empty. (10 min)
* Updating Manual. (30 min)

February 16, 2020:

* Updating manual and screenshots for final submission (1 hour)

**Total: 10 hours**

**FULL TOTAL: 39.5 hours**

# 5 Compiling Instructions

A MakeFile is included in the submission. The program is compiled using C++11 compiler. Once in the folder directory, enter “make” to build the project using the command line. The executable named “goStop”, builds on the same directory. The game can be initialized from a previous configurations file or as a new game.

# 6 Program Screenshots:

A screenshot of a cell phone

Description automatically generated

**Screenshot 1:** User Turn with help mode with each user input. Picks card from hand according to recommendation and adds to layout to create a stacked pair. Then draws a card from the stock pile, which does not have any match so it is added to the layout. The capture is made and added to capture pile to make a complete set of 4.

A close up of a piece of paper

Description automatically generated**Screenshot 2**: Game setup and computer’s play turn.

A screenshot of a cell phone

Description automatically generated

**Screenshot 3**: End game results.

A screenshot of a cell phone

Description automatically generated

**Screenshot 4**: Loading and saving the game.

Thank you