***California State University, Long Beach***

**CECS 282** *C++ for Java Programmers*

Final Project: Blackjack Slot Machine

December 15, 2017

Steven Garcia & Gabriel Flores

Introduction

The purpose of this program is to simulate a Blackjack Slot Machine. The user is prompted for an account number and the amount of money to bet. Two user cards will then be selected randomly and will be shown to the user. Another two cards will be randomly selected for the dealer’s hand. The program will display the value of the user’s cards and will prompt one of the following actions: hit, stand, or split. If the value of the user’s cards is less than or equal to 21 and worth more than the dealer’s, the user receives double the betted amount. If the values of the cards are the same, there is a tie and the money is split. Otherwise, the user wins nothing. The user can choose to continue playing after each hand.

Program Analysis

Our final project consists of Player, Account, Hand, and Card classes. The Player class represents a player in the blackjack game, the Account class represents the Player’s account, and the Hand class encapsulates the Card class to represent the Player’s blackjack hand.

Player class has get and setter methods for accessing attributes for their corresponding Account and Hand objects. The Player class also has a bet() method to bet a certain amount, a stand() method to stand, a hit() method to hit and get another card, and a split() method to split his or her hand. The usingSplitHand variable is a Boolean that indicates whether to add a card to the primary hand or the secondary hand. The player also has methods winGame(), loseGame(), and tieGame() to indicate whether or not he or she wins.

The Account class has get and setter methods for its attributes. It has private variables accountNumber and totalAmount to store the account number and amount in the account.

The Hand class has private variables cards, valueOfCards, numberOfCards, and cardMemoryCapacity. It stores the Card objects by having the cards variable point to a dynamic Card array. It has the appropriate get and setter methods to retrieve and manipulate the value and number of cards. It also has the allocateDynamicMemory() function to handle a possible memory leak.

The Card class has the get and setter methods to access the cardType and value. The value of the Card is determined by the cardType and is based off of the traditional set of 13 cards.

The program starts by asking the user for his or her account number and the amount of money to bet. Afterwards, a Player object is created for both the user and the dealer. Then the program executes a while loop that begins by dealing two cards to both the user and the dealer. The cards are randomly generated by the generateRandomCard() function. Both the user’s and dealer’s total value from the cards are output to the screen so that the user is aware of the current cards. After dealing the initial cards, another while loop is executed, which prompts the user to enter a move using the getUserDecision() function. The dealer is set to always hit until the value of his or her cards is above 15. If either the user’s or the dealer’s cards go over 21, the other player wins and the user is prompted to play again with the playAgain() function. If the user decides to play again, the loop reiterates. Otherwise, the loop breaks and the program ends.