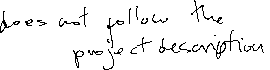
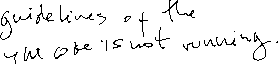
Tyler Pranger



Professor Arias



Software Development I

26 April 2019

Blackjack is a popular card game, played frequently in casinos for gambling, or in households for leisurely fun. It can be represented in physical form, with playing cards, but I will be attempting to create a game of Blackjack with the help of Java code. To do this, I will be using 5 different classes, these being the main function for the game, the class that handles the deck, the class that gives the cards their value, a class that will manage the cards within the player’s hand, and an extension of that class that will give the cards their point value and position the cards within the deck. The reason why I want to do this project is because of my interest in gaming as a whole. I want to be able to figure out how coding a game works, and using a physical card game that I have played often throughout my life, I believe that I can achieve this goal.

The main elements of a game of blackjack are the cards themselves, the freedom to draw cards until the player wishes to hold onto his hand, and the concept of wagering money so that the game has a sense of risk and reward. All of these elements will be captured using each of the classes, assigning values to an entire deck of 52 different cards, each with their own suit and point value. The player will start out with a certain sum of online currency, which will start at $200 dollars but can be changed directly within the code. If the user goes beyond 21 points, they will automatically lose the round, as in a normal game of Blackjack, and this will be repeated for the dealer’s turn. All of these elements have to be represented within the program so that the player feels like he is playing an actual game of blackjack without the barrier of a digital screen.

Every class that is part of this project is vital for the Blackjack program to work. These classes function off of each other, as the Card class is what assigns values to the digital cards. The card class contains two strings, with one string representing the numbers and face cards, and the other string representing the suits. For each suit, the cards are given values equaling one to eleven, as these are the card values that will be used to play a game of blackjack. For example, in the overall code, the player will be given a random card, such as the ten of hearts and the four of spades. The player will know that these particular cards have set values that total up to fourteen, and this is because of the Card class. In the Deck class, an array list is created in order to store the cards for a new, shuffled deck. The cards created from the Card class are stored into this array in the Deck class, and then another function takes the values in the array and randomizes them. For the Hand class, all of the card values are sorted together so that matching suits are kept together in the player’s hand. This class also manages how many cards are in the hand, removes the cards from the hand, and positions the cards in the deck after it is shuffled. The Dealer class was created as an extension to the Hand class, and it is responsible four counting up the cards dealt after every hit or pass, as well as counting up the total points that the player and dealer have within their hands. Finally, the main function, called Blackjack, is the class that would allow the user to actually play the game. The class provides the user with an assigned amount of starting money, which can be used to wager against the dealer every round. The program is designed to allow the player to keep playing until they either run out of money to bet with, or decide to exit the program with the money they have made. This program will also, as to be expected, go through an entire game of Blackjack with the player. It will deal out cards using methods from the Card and Hand classes, prompt the user with questions about whether they want to add more cards into their hand or keep the hand they were dealt, and also enact the entire turn of the dealer, allowing him to collect more cards to try and beat the player’s hand. If either player gets over 21 points in their hand, the program will automatically send the player to the method for a game over, either collecting money from the dealer or losing the money that was bet. I have looked into other programs of famous card games, such as poker, and the classes within those programs work in a similar way, creating the cards in one class and having a separate class to shuffle the whole deck.

Creating this program will allow the user to play a fully-fledged game of blackjack. The only code that can be edited by the user before starting would be how much money the user can start with, although the player should naturally know a reasonable amount of money to have. Everything will be formatted using input commands from the user to play the game. In conclusion, the program should be able to take the player through as many games of Blackjack as the user desires, stopping if the user prompts a certain command in the main function. Working on this project should help me realize what is takes to create a functional game.

|  |
| --- |
| Hand |
| -hand: <Card> |
| +Hand()  +clear(): void  +addCard(Card c): void  +removeCard(Card c): void  +getCardCount(): int  +getCard(position: int): Card  +sortBySuit(): void  +sortByValue(): void |

|  |
| --- |
| Dealer: extends Hand |
| +getBlackJackValue(): int |

|  |
| --- |
| Card |
| -suit: int  -value:int  +Spades, Hearts, Diamonds, Clubs: int  +Ace, Jack, Queen, King: int |
| +Card(rank: int, Suit: int)  +getSuit(): int  +getValue(): int  +getSuitForString: String  +getValueForString: String  +toString(): String |

|  |
| --- |
| Deck |
| -numCards: int  -deck: Card[] |
| Deck()  -fill(): void  +shuffle(): void  drawCard(): Card  dealCard(): Card |

Bibliography

Arnold, Ken, James Gosling, and David Holmes. *The Java programming language*. Addison

Wesley Professional, 2005.