Casino Royale APCS1 pd 1
Yuanchu Liu (Leo), Michael Ruvinshteyn, Andrew Qu 2017-01-09

Final Project Proposal

Casino Royale

We plan to create a collection of common casino games, aiming for a slot machine, Blackjack, Roulette, /*and poker*/ (adding more if we have the time). In addition to the driver file (Woo), we will create the superclass Game, from which the subclasses Blackjack and Roulette and etc. extend. Each Game will have instance variables to track the value of the player's current bet and how much their money has changed since sitting down at a new game (if the user plays 5 games of roulette, it records the change in the user's balance from the start of the first game until now).

The amount of money that a player has is kept in the driver file, and is delivered to each game, with amounts won or lost updated within each game like mentioned above, and returned to be stored within the driver file.

Array<String> of length 52 in Game to represent a deck of cards(?)
The first character will represent the card's suit (D, C, H, S). This is followed by a number from 1-13, representing aces, number cards, and then the face cards (J, Q, K). A card's value can be determined a number of ways, using substrings and parseInt()

Blackjack:
-Since blackjack doesn't use suits, and treats all face cards as 10, it could be easier to instead generate an array of integers from 2-11, with four times the number of 10s. However, the more general deck could be used for other card games that do use suits
-A method to simulate drawing the top card of the deck. Essentially, it'll work by processing the first element to determine its value (in this case, cutting off the first character since suits are ignored, and using parseInt() on the remaining string. If the value is >10, meaning it was a face card, the value 10 is used instead). This element is then removed from the array
-Instance variables to track player's total card value and that of the dealer
-A lot of information will be printed to the terminal, including the cards in your hand, their total value, a prompt asking the player to input whether they want to "hit" or "stand" and of course messages that notify whether they've won or not and their winnings/losses
-The player will also have to place a bet before each game of course. Their current balance and bet will always be printed when applicable
-Possible extra features: option to make aces high or low, option to "count cards" such that the number of cards that have been dealt are displayed, ability to play against an AI dealer instead of a 2nd human player

-The AI will not have to be too complicated. The dealer will simply “hit” if their total card value is lower than the player’s, since the dealer would lose otherwise

Roulette:
-The player will choose whether to bet on a number (bigger payoff) or a color (smaller payoff). The amount the user bets will be multiplied by a constant (tbd) depending on the previous specifications, as well as the probability of their choice (i.e. green has a smaller chance than red and black, and will thus receive a higher multiplier).
-After putting the amount in, the program will “spin” the roulette wheel. If the user wins, they receive the amount they put in plus that amount times the multiplier.

Poker:
-Uses a complete deck represented by an String[] length of 52, with a counter for number of cards already dealt.
-Shuffling would be done by running swap on the String[] multiple times similarly to how it was done in Slots.java.
-Dealing would be assigning the String representation of a card in deck in position <counter> in order to ensure no repeats are dealt and remaining cards are being tracked.