CSE1322L Assignment 1 - Spring 2023

Introduction:

To test your knowledge of CSE1321, you'll be writing a game for your first assignment. Specifically you'll implement a poker game. You won't have to deal with betting and some other things for this assignment, we are going to simplify it for you.

Everything you do in this game will be in text, so you'll be printing lots of stuff to the console. We'll use extensively ArrayLists(java) or Lists(C#), and you'll need lots of loops, if statements and variables along the way. You'll use classes that we provide, and you'll write your own class with methods.

Playing Cards:

We will use a standard US playing card deck. Each card has a value (e.g. 3) and a suite (e.g. Hearts). The cards have a value which starts at Ace, then 2, 3, 4, 5, 6, 7, 8, 9, 10, then Jack, Queen, and finally King. They are usually abbreviated as A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K.

Cards also have a suite. There are 4 suites: Clubs (♣), Diamonds (◆), Hearts(❤) and Spades (♠). For this implementation we won't use the symbols, we'll just type out the suite name.

We are providing you with a class called PlayingCards which understands this, and can shuffle the deck and tell you what the first card is.

How Poker is played:

In this implementation there will be exactly 2 players. The game begins by dealing 5 cards to each player. One card is dealt to player 1, then the next card is dealt to player 2. The next card goes to player 1,and this pattern continues until both players have 5 cards.

Players then look at their hand, and figure out how strong their hand is. The person with the strongest hand typically wins (betting may change that, but we are not dealing with betting in this implementation)

The strength of a hand is determined by the cards in the hand. There are established rules which are explained below. All hands (5 card combinations) will result in one of the following hand strengths:

Possible Poker Hands (By strength):

Here are the possible hands you can get from strongest to weakest.

- Royal Flush - When a player has exactly A K Q J 10 of the same suite. There are 4 versions of this hand (due to the four suites)
- Straight Flush - When a player has five cards that are sequential and of the same suite. A royal flush is a specific straight flush. For example 9 10 J Q K of Hearts, or A 2 3 4 5 of Clubs would be straight flushes.
- Four of a kind - When a player has all four of a single value. For example 2 2 2 2, or K K K K. They will inherently be of the 4 different suites.
- Full House - When a has three cards of one value and two cards of another. The suite is irrelevant.
  For example A A A K K would be a full house, as would 2 2 2 3 3.
- Flush - This happens when all cards in a player's hand are of the same suite.
  For example, all clubs, all diamonds, all hearts or all spades
- Straight - This happens when the values of the cards in a hand are sequential.
  For example 8, 9, 10, J, Q. Or A, 2, 3, 4, 5. The suites don't matter, just the values. Note that the A can be the lowest card in the deck or the highest.
  A 2 3 4 5 is a straight, as is 10 J Q K A.
- Three of a Kind - This happens when a player has three cards of the same value.
  For example three 8's. e.g. 2 2 2 K J
- Two Pair - This happens when you have two cards of one value, and another two cards of a different value. For example 8 8 2 2 A, or K K A A 3
- Pair - A pair happens when you have two cards that are the same value. For example you have two 8's or two A's in your hand. If multiple people have a pair, the person with the highest value pair wins.
- High card - Whoever has the highest value card wins (unusual this would win, but technically possible)

Tasks:

First you will need to download two files from the FYE site (next to this doc). If you are a C# student you'll want PlayingCards.cs and main.cs. If you are a Java student you'll want PlayingCards.java and Main.java. You'll want to import (or copy/paste) these into your IDE to start.

Your task is to write a new class called **Poker**. It will need a number of attributes and methods which are explained below:

- First create a private attribute of type PlayingCards. Call it deck
- Create two ArrayLists (Java) or Lists (C#) called hand1 and hand2. These should be of type string as each card is a string.
- Write a method called dealHands(). It should take no parameters and return void. Take the first card off the deck and put it in hand1. Then take the next card off the deck and put it in hand 2. Do this until both hands have 5 cards.
- Write a constructor that takes no parameters
    - Instantiate a PlayingCard object on your deck variable.
    - Call the shuffle() method in PlayingCard to shuffle the deck.
    - Call the dealHands method you just wrote.
- Write a constructor that takes in two ArrayLists (Java) or Lists (C#).
    - Instantiate a PlayingCard object on your deck variable.
    - Set hand1 to the first parameter, and hand2 to the second parameter.
    - This constructor is used to test your later code.
- Write a method called showHand which takes in an integer and returns void. If the integer is 1, show hand1, otherwise show hand2
    - Print out "Player 1's hand:"
    - Print out each of the cards in hand1. Then print an empty line.
    - Print out "Player 2's hand"
    - Print out each of the cards in hand2
    - See sample output below.
- Write a method called countSuite, it should take in an ArrayList (Java) or List (C#) of strings (the hand). It should return an array of integers.
    - Go through all the cards in the hand. For each card, extract the suite from the string (you might need string split for this). Count how many cards are clubs, and put that value into cell 0 of the array you are going to return. Count how many cards are diamonds, and put that value into cell 1 of the array. Count the Hearts and put that into cell 2. Finally count the spades and put that value into cell 3. Return the array of integers which has the count of how many each suite you have. This will be useful when you check for flushes.
- Write a method called countValues. It should take in an ArrayList (Java) or List (C#) of strings (the hand). It should return an array of integers.
    - Create an array of size 14.
    - Extract the value out of each card in the hand.

    - Use the array to keep track of how many of each value you see. For example if you have an Ace (A) increment cell 1 of the array. If you have a card with a 6 you'd increment the value in cell 6 of the array. Increment cell 11 for Jack (J), cell 12 for Queen (Q) and cell 13 for King (K).
    - Return the array.
- Write a method called numPairs which takes in an array of integers (the one you got back from countValues). It will return an integer.
    - Look through the array for any cell which has a 2 in it. Each cell that has a 2 indicates you have a pair. Return the number of pairs you found.
- Write a method called threeOfAKind which takes in an array of integers (the one you got back from countValues). It returns an int. It should look through the array and see if any cell has a 3 in it, indicating there are 3 of that value. If so, return the cell number that has the 3 in it. Otherwise return 0.
- Write a method fourOfAKind which works the same as threeOfAKind but looks for a cell with a 4 in it.
- Write a method called fullHouse. It should take in an array of integers (the one you got back from countValues). It returns a boolean. Check if you have threeOfAKind and one pair. You have methods for checking each of these conditions. If yes, return true, otherwise false.
- Write a method called straight. It should take in an array of integers (the one you got back from countValues). It returns a boolean.
    - You'll need to see if there are sequential cells in the array with a 1 in them. This would indicate that you have a straight. You'll also need to check for the one odd straight where you have a 10, J, Q, K, A. This is odd, because the 10 is in cell 10, J in 11, Q in 12, K in 13, but the A is in cell 1.
- Write a method called flush. It should take in how many clubs, diamonds, hearts and spades you have in the hand. If any of those values are 5, return true, otherwise false.
- Write a method called straightFlush. It should take in an array of integers (the one you got back from countValues) as well as the number of clubs, diamonds, hearts and spades in the hand. Then simply check if you have both a straight and a flush. You have methods for each. If they are both true, return true, otherwise false.
- Write a method called royalFlush which takes in the array of integers you got back from countValues, followed by the number of clubs, diamonds, hearts and spades you have. This method should return a boolean (Java) or bool (C#).
    - If you have 5 clubs, 5 diamonds, 5 hearts or 5 spades and cells 10, 11, 12, 13 and 1 of the array have a 1 in them, you have a royal flush, otherwise you don't.
- Write a method called scoreHand which takes in an integer and returns a string.
    - The method will return a string which contains the strength of the hand.
    - If the passed in parameter is 1, this method will return the strength of hand1, otherwise it'll return the strength of hand2.
    - To determine the strength, start at the strongest hand (royal flush) and check if the hand qualifies, then try the next strongest. If none qualify you'll return "High Card".

Example Runs: [User input in red]

* Note: The first five hands should return exactly the same output, but the sixth hand is a random hand, which will be different each time you run the code.

Player 1's hand:
A of Hearts,K of Hearts,Q of Hearts,J of Hearts,10 of Hearts,
Royal Flush
Player 2's hand:
9 of Hearts,K of Hearts,Q of Hearts,J of Hearts,10 of Hearts,
Straight Flush

Player 1's hand:
9 of Hearts,9 of Spades,9 of Clubs,9 of Diamonds,2 of Hearts,
4 of a kind
Player 2's hand:
9 of Hearts,9 of Spades,2 of Clubs,2 of Diamonds,2 of Hearts,
Full House

Player 1's hand:
2 of Hearts,9 of Hearts,A of Hearts,3 of Hearts,7 of Hearts,
Flush
Player 2's hand:
5 of Hearts,8 of Clubs,9 of Diamonds,7 of Spades,6 of Hearts,
Straight

Player 1's hand:
A of Hearts,A of Clubs,A of Spades,3 of Hearts,7 of Hearts,
3 of a kind
Player 2's hand:
A of Hearts,A of Clubs,7 of Spades,3 of Hearts,7 of Hearts,
2 pairs

Player 1's hand:
A of Hearts,A of Clubs,7 of Spades,3 of Hearts,K of Hearts,
1 pair
Player 2's hand:
A of Hearts,4 of Clubs,7 of Spades,3 of Hearts,K of Hearts,
High Card

Player 1's hand:
10 of Spades,6 of Clubs,Q of Clubs,K of Diamonds,4 of Hearts,
High Card
Player 2's hand:

9 of Spades,2 of Clubs,K of Spades,3 of Hearts,5 of Spades,
High Card

Submitting your answer:

You'll submit just Poker.java or Poker.cs on gradescope as assignment 1. You do not need to turn in the 2 files we gave you.

Please follow the posted submission guidelines here:
https://ccse.kennesaw.edu/fye/submissionguidelines.php

Ensure you submit before the deadline listed on the lab schedule for CSE1322L here:
https://ccse.kennesaw.edu/fye/courseschedules.php