B20_Poker_Projecet

The purpose of this project is to create a Poker Game using OOP concepts
The poker game variation used for this project is texas holdem. So all the game rules are based on this version (reference video provided at the end after game rules)
If you are totally new to poker, i suggest read through first 3 point and watch the example video, if not can just jump into 3rd page to the actual project

1. Poker general Intro

- is a card game, will be played on deck of 52 cards
- each player will have 2 cards
- And 3 cards will be placed on table, and as game goes on 2 more will be added
    - So there will be 5 card at the table at the end
    - Cards on the table will be referred to as community card
    - Community cards are visible to everyone
- Objective of the game is to pick 3 cards from the community cards and combine it with the given 2 cards, and highest rank of those combined 5 cards will win
    - The rank will be explained below
    - Community cards are shared by everyone, so each player can pick any 3

1.1 Playing Card Intro

- A deck of cards contains 52 cards
- Each card contains 3 specific information
    - Value: 2, 3, 4, 5, 6, 7, 8, 9, 10, J, K, Q, A
    - Suit type: Spade (♠), Club(♣), Heart(♥), Diamond(♦)
    - Colours: Black and Red
        - However in this project we won't be dealing with colours
- Example of a Card
    - 3 Spade, 3 ♠, 3 S
        - All represent same card: 3 of Spade
        - Pick one to use for this project

2. Card Ranks

- In texas holdem poker, based on the combination of 5 cards(2 given, 3 community card), there are 10 ranks
- The ranked given here are listed from lowest rank to highest rank
    - Rank 10: High card (all unrelated cards)
    - Rank 9: One pair (one paired cards)
    - Rank 8: Two pair (two paired cards)
    - Rank 7: Three of a kind (3 cards are the same)
    - Rank 6: Straight (all 5 cards are consecutive)
    - Rank 5: Flush (all cards have the same suit type)
    - Rank 4: Full House (3 same card and 1 pair)
    - Rank 3: Four of a kind (4 same Card)
    - Rank 2: Straight flush (Straight with same suit type)
    - Rank 1: Royal flush (Straight flush but also contain Ace)
- Reference Picture: https://ibb.co/17MRKzM

3. Game Rules

- More in depth guide will be provide later, As intro to this project, this should be the game flow for starter
    - A deck of Card will be created, that contains 52 cards
    - Those 52 Cards will be shuffled
        - Position of the cards will be randomized
    - Each player will be given 2 cards from those shuffled deck
        - All taken out cards should be in order
        - For simplicity, let's say we will have 3 player for now
    - Another 3 cards will be taken out as community card
    - Each player card will be combined with the community card
    - Rank will be given for each players
    - Ranks of the each player will be compared, and highest rank will be the winner

- This is an example of how it would look, let's say i generated 10 Cards, for simplicity i will just use their value for now
    - 2, 3, 5, 6, 8, 4, 6, 7, 10, 6 (Cards are generated)
    - 5, 2, 4, 3, 6, 6, 10, 6, 2, 7 (Cards are shuffled)

    - Player1: 5, 2 (each cards are taken out in order)
    - Player2: 4, 3 (very important)
    - Player3: 6, 6
    - Community: 10, 6, 2

      (player cards And community cards are combined together)

    - Player1: 5, 2, 10, 6, 2 One pair rank 9
    - Player2: 4, 3, 10, 6, 2 High Card rank 10
    - Player3: 6, 6, 10, 6, 2 Three of A kind rank 7

    - Player3 Wins because he had the highest rank card

- In short, this is how it suppose to work, However, in reality it is gonna get really complicated and more will be discussed later

- Reference video:
    - https://www.youtube.com/watch?v=NlFguTSypBQ

4. Custom Class for the project

This is where the fun Begins, At this point I assume everyone is familiar with the general term of poker and has the basic understanding of how it works. So here what we are trying to do is create multiple classes, whereas each will contain some specific set of attributes, so that when we create an object using that class, we could easily call out those values. Currently we are planning to create 5 separate Custom classes: Card, Hand, Players, Deck, Utility
Custom class names are not set, can always change it to something more meaningful, Naming is hard!

Terminology:

Card: contains 2 information: card value and Card suit type
Hand: will contain multiple Card objects, typically 5 cards for this project
Players: contains multiple Hand objects

Think of them as Arrays,
Card will be 1 dimensional array {2, “S”} -> 2 is value, S is suit type -single card
Hand can be 2 dimensional array {{2,”S”}, {3,”D”}, {K,”S”}, {10,”H”}, {7,”D”}}; -contains 5 cards
Players will be 3 dimensional array{{{6,”S”}, {5,”D”}, {K,”S”}, {10,”H”}, {7,”D”}},
{{2,”S”}, {3,”H”}, {Q,”H”}, {9,”H”}, {5,”S”}},
{{8,”D”}, {6,”D”}, {K,”S”}, {10,”H”}, {A,”D”}}}; -contains 3 hands

However, we won't use any multi-d Arrays here so don’t worry. My point is by using those custom classes we can have direct access to the value of each element. For our project it will look like something like this

Card test = new Card();
Hand {new Card(), new Card(), new Card(), new Card(), new Card()};
Players {new Hand(), new Hand(), new Hand()}

Just for an example, so that is the main idea, hope it make sense, if not let's try to break it down

Custom Class #1: Card

Let’s start with an easy class first, this class will represent single Card only, so only the attribute we need from this class is Card Value and Card Suit Type:

Attribute:

String CardValue;
String CardSuitType;

Constructor with 2 parameter: (String value, String Type)

Since each Card always has to contain some information, i think it will be easier when we just provide that information when we created a Card Object, hence i suggested using 2 parameters instead of default Constructor.

Custom Class #2: Hand (will probably need better name)

Now, it will get a bit more complicated. Hand is basically a class that contains multiple Card Objects. So here it will contain slightly more information.

First of all, since we can create multiple Card objects, and since it is an object, we can store them in an Arraylist, it will work as such: ArrayList <Card>;
ArrayList<Card> listofCards = new ArrayList<>();
and like that we can create an empty Card ArrayList
one of the very important concepts for this project. Just like any other Arraylist we have learned before, it will work the same. All the ArrayList methods will work for it as well. So please keep that in mind.

Now back to the Hand class:

Constructor (ArrayList<Card> listOfCard)

We will also use an custom constructor here as well, since in order to create a Hand object, we need list of Card objects

Attributes:

String name;
ArrayList<Card> cards;

boolean isFlush;
boolean isStraight;
int rank;

Optional:
ArrayList<String> valueOfCards; (contains string value of all the given Cards)
ArrayList<String> typeOfCards; (contains suit type of all the givens cards)

Methods:
As many as needed;

The main purpose of this Class is to get the rank value for this given Hand, so in order to find the rank, use as many custom method as needed, also any needed Attributes can be added as well, as long as it works;

Side note: in theory this class can contain as many Cards as given, since the requirement is just an ArrayList<Card>, however, since in poker, we are comparing 5 cards only, so try to keep the size of the arraylist to 5, will make life much easier.

Custom Class #3: Players

This is the toughest part of the whole poker game. Players class contains multiple Hand objects, So purpose of this class is to figure out the winner of the game. Since we already figured out the rank for each Hand, we will return the highest rank Hand as a winner. However, the main problem comes from when the ranks are equal, so we have to figure out which Hand will win. For example: we have 3 hand objects with 1 rank 10 and 2 rank 9, since we have 2 rank 9 (one pair), we have to compare which has the highest pair etc, so it gets complicated.

Constructor (ArrayList<Hand> listOfHand>
Same as Hand class, the requirement to create this object is list of hand, basically the cards that each player holding

Attributes:
Int size (returns arraylist size, basically how many players are there)
ArrayList<Hand> hands; (hand of each players)
ArrayList<Hand> equalRanks; (hand of players that are equal rank)
(only need to compare equal rank hand instead of all the player

String [] rankArr; (return the rank for each hands)
Hand bestHand; (most important: return the winning hand)
(we can get the rank for this by just using bestHand.rank from previous class)

Methods:
As many as needed

Same as previous class, here at the end of the day, we are trying to find the winner's hand, so as long be able to find it, that is fine. since each person will have a different way of solving it

Custom Class #4: Decks

Purpose of this class is to generate 52 cards and be able to shuffle it
Also for all the card object, the required information are came from this class
So basically, at the end, this class should return ArrayList<Card> that has 52 card object in it
More will be added later for this class

Custom Class #5: Utility

Whatever left off, can be placed in here

For now, this is where we are at, we are far from done, once more people are on the same page, we can keep continuing and add more functionality in the future