Software Design CS 215: Homework # 2 Mina Khan

(see UML class diagram on last page)

CRC:

Class: GameInterface	
Responsibilities: - knows about the status of the game, i.e. players' score, players' moves - display the graphics related to the game	**Collaborators**: - GameManager

Class: GameManager	
Responsibilities: - starts the game and 'plays' the game - to 'play' the game, there are several helper methods to keep track of moves of all players, determine winner and evaluate player moves - keeps track of the game status and game end	**Collaborators**: - Pot - GameInterface - Player - Deck

Class: Player	
Responsibilities: - keeps track of player's bets, cards, chips and moves - I think each player would have 21 (i.e. 7 choose 5) hands because they can create a hand of five cards from their 7 possible cards	**Collaborators**: - GameManager - Hand - Move

Class: Hand	
Responsibilities: - this is a set of five cards. At the end of the game, each players tries a different combination of community and hole cards to create a set of five cards that gives them the highest hand_rank	**Collaborators**: - Hand_Rank - Card

Class: Deck	
Responsibilities: - keeps track of all cards in the deck and shuffles and deals cards to players	**Collaborators**: - Card

Class: Hand

Responsibilities:	**Collaborators**:
- has a set of five cards - it sorts the cards to evaluates the hand_rank	- Card - Hand_Rank

Class: Card	
Responsibilities: - keeps card of the suits and rank of the card - compares cards - keeps track of whether or not the card is flipped	**Collaborators**: - Suit - Rank

Class: Rank	
Responsibilities: - enumerates all ranks	**Collaborators**:

Class: Suit	
Responsibilities: - enumerates all suits	**Collaborators**:

Class: Hand_Rank	
Responsibilities: - enumerates all hand_rank	**Collaborators**:

Class: Move	
Responsibilities: - enumerates all moves	**Collaborators**:

• (10 points) Discussion of design tradeoffs

There were a couple of things that I considered adding, for example a Dealer class, so that a dealer is different from a player, but then the dealers keep rotating and players switch from being dealers to players. Therefore, I decided to keep track of this behavior using a Boolean in player class, instead of a dealer class.
I know that there are different kinds of chips in poker so I could have created a separate class for that, but the information was included in the Texas Holdem Poker description so I didn't go in the details of chips in my design.
I used enumerated types a lot because they allow convenient use of switch statements and I can specify the values of enum constant at the creation time. Move, Hand_Rank, Suit and Rank are enum in my design.
In order to make hand ranking easier, I have sorting methods in hand to make ranking hands easier. I also have a compare method for cards so that we can compare cards efficiently.

GameManager is my main class and it is used for starting and ending the game. This class has a number of private helper to help in the smooth running of the game.
I didn't see a lot of opportunities to use inheritance, but I made sure that the design was modular and all the functionality pertinent to a class was taken care of in the relevant classes. I used abstraction to make sure that none of my classes were "God classes", for example, I created a Hand class which deals with ranking the hands, so that neither the player, nor the GameManager would have to rank hands.
I have tried to minimize coupling between classes so that classes are connected to classes that they use. Although I tried to minimize coupling and maximize cohesion, I believe that may be a better way of reorganizing my classes so that I use inheritance and further minimize coupling and maximize cohesion. Nonetheless, I have created the best UML design I could come up with considering all the class design principles we have learnt in class.