Project #2. A Card Game

Handed out: Tuesday, 11/29/16	Due date: Tuesday, 12/6/16

Submission instructions

Inside that the homework subdirectory, create a directory for project #2, and call it `project2`. When you finish the assignment, go to the homework directory and submit it as follows:

```
$ submit movery project2 project2
```

What to do:

1. Create a program to shuffle and deal a deck of cards. The program should consist of class `Card`, class `DeckOfCards` and a driver program. Class `Card` should provide:

a) Data members face and suit of type int.

b) A constructor that receives two ints representing the face and suit and uses them to initialize the data members.

c) Two static arrays of strings representing the faces and suits.

d) A toString function that returns the Card as a string in the form “face of suit.” You can use the + operator to concatenate strings.

Class `DeckOfCards` should contain:

a) A vector of Cards named deck to store the Cards.

b) An integer currentCard representing the next card to deal.

c) A default constructor that initializes the Cards in the deck. The constructor should use vector function push_back to add each Card to the end of the vector after the Card is created and initialized. This should be done for each of the 52 Cards in the deck.

d) A shuffle function that shuffles the Cards in the deck. The shuffle algorithm should iterate through the vector of Cards. For each Card, randomly select another Card in the deck and swap the two Cards.

e) A dealCard function that returns the next Card object from the deck.

f) A moreCards function that returns a bool value indicating whether there are more Cards to deal.

The driver program should create a DeckOfCards object, shuffle the cards, then deal the 52 cards.

2. Modify the program above so that it deals a five-card poker hand. Then write functions to accomplish each of the following:

a) Determine whether the hand contains a pair.

b) Determine whether the hand contains two pairs.

c) Determine whether the hand contains three of a kind (e.g., three jacks).

d) Determine whether the hand contains four of a kind (e.g., four aces).

e) Determine whether the hand contains a flush (i.e., all five cards of the same suit).

f) Determine whether the hand contains a straight (i.e., five cards of consecutive face values).

3. Use the functions above to write a program that deals two five-card poker hands, evaluates each hand and determines which is the better hand.

4. Modify the program above so that it can simulate the dealer. The dealer's five-card hand is dealt "face down" so the player cannot see it. The program should then evaluate the dealer's hand, and, based on the quality of the hand, the dealer should draw one, two or three more cards to replace the corresponding number of unneeded cards in the original hand. The program should then reevaluate the dealer's hand.

5. Modify the program you developed in #4 so that it handles the dealer's hand, but the player is allowed to decide which cards of the player's hand to replace. The program should then evaluate both hands and determine who wins. Now use this new program to play 20 games against the computer. Who wins more games, you or the computer? Have one of your friends play 20 games against the computer. Who wins more games? Based on the results of these games, make appropriate modifications to refine your poker-playing program. Play 20 more games. Does your modified program play a better game?

Specification:

Please include a comprehensive README file that explains the details of your implementation. Your README should include a description of the strategies you implemented in #5 and report the results of your two 20-game experiments.

Please insure that your code is well-commented.

Extra Credit:

Extra credit will be applied to the final score for implementing a "smart" strategy to determine whether to draw more cards and how many, in order to replace existing cards. For example, you may consider what cards have been dealt and what you have at hand, to see the likelihood you will get a flush or straight, etc.