Creating a Deck of Cards (10 Points)

Create a Standard Deck of 52 cards. Each suit (Clubs, Diamonds, Hearts and Spades) and each card face (Deuce… Nine, Jack, Queen, King and ace).

Create the implementation code for the following UML definitions for the Deck and Card classes:

```
Deck

- theDeckOfCards :  vector < Card >

- indexOfNextCard : size_t


+ Deck (  )

+ shuffleTheDeck( ) : void

+ dealOneCard( ) : Card

+ cardsLeft( ) : bool


Card

- face :  size_t

- suit : size_t

- namesOfFaces[ ] : static const string

- namesOfSuits[ ] : static const string


+ Card ( cardSuit : size_t, cardFace : size_t )

+ getFaceValue( ) : size_t

+ getSuitValue( ) : size_t

+ cardToString( ) : string
```

Each class needs its own interface file (.hpp file with prototypes), and an implementation file (.cpp file with code for each member function). card.hpp / card.cpp

+ deck.hpp / deck.cpp Each interface must be strictly adhered to for credit. Same names and functions as defined here.

You will need to create a class for a single card which has:

- Private data for the face and suit of the card
- A constructor to initialize the card (face and suit)
- A getFaceValue() function (Note there is No Set…you can't change a card)
- A getSuitValue() function (Note there is No Set…you can't change a card)
- A cardToString() function to print out the string representation of a card.

Then create your main( ) (which should be less than 10 or 15 lines of code) so that it :

- Instantiates (creates) the Deck of cards.
    - The Deck class will use a Vector. In main is it just a variable like "yourDeck".
    - The Deck class will use the emplace_back member function to add each card to the deck.
- Shuffles the Deck of cards
    - The Deck class will use the random swap method defined in the book.
- Deal the entire deck of cards
    - main must display them as Card Face and Suit as a string (use the dealOneCard( ) and CardToString( ) member functions).
    - setw(20) and a counter can help you do the columns.
    - Note: The cardsLeft( ) function tells you when to stop dealing.

Sample Output:

```
Nine of Clubs       Seven of Spades     Five of Hearts      King of
Hearts

Six of Clubs        Nine of Spades      Six of Hearts       Ace of
Clubs

King of Clubs       Seven of Hearts     Eight of Hearts     Five of
Spades

Deuce of Clubs      Ace of Spades       Seven of Diamonds   Jack of
Hearts

Three of Spades     Queen of Spades     Nine of Hearts      Deuce
of Hearts
```

```
King of Spades     Ten of Clubs       Four of Clubs      Ace of
Diamonds

Ten of Hearts      Four of Hearts     Eight of Spades    Five of
Clubs

Deuce of Diamonds  Ten of Diamonds    Four of Diamonds   Queen
of Hearts

Three of Hearts    Three of Clubs     Jack of Clubs      Eight
of Diamonds

Ten of Spades      Four of Spades     Seven of Clubs     Eight
of Clubs

Queen of Clubs     Nine of Diamonds   Deuce of Spades    Jack of
Diamonds

Six of Spades      Six of Diamonds    King of Diamonds   Queen
of Diamonds

Jack of Spades     Three of Diamonds  Ace of Hearts      Five of
Diamonds
```

Submit all 6 files (card.hpp, card.cpp, deck.hpp, deck.cpp, main.cpp and output.txt)