Larry Nguyen

4/15/20

CECS 130-1

Final Project Report

```
8  class Poker{
9  public:
10     int bet;
11     int chips;
12     int hands;
13     int wins;
14     int loss;
```

Creating class for poker.

```
16     Poker(){
17      chips=100;
18      hands = 0;
19      wins = 0;
20      loss = 0;
21      bet=0;
22     }
```

Constructor to initialize chips as $100 and the others at 0.

```
21     string randomCard(){
22      srand(time(NULL));
23      int number = rand() % 9 + 1;
24      stringstream its;
25      its<<number;
26      string suit[4]={"C","S","H","D"};
27      int suit_index= rand() % 3 + 0;
28      string card = its.str()+suit[suit_index];
29      return card;
30     }
```

Function to generate a random card. Seeding srand and time to seed random with different seeds. Uses rand to generate numbers from 2 to 10. Converts number to stringstream. Generates random suits.

```
vector<string> hand(){
    hands++;
    vector<string> cards;
    for(int i=0;i<4;i++){
        while(1){
            string card=randomCard();
            if(find(cards.begin(),cards.end(),card) == cards.end()){
                cards.push_back(card);
                break;
            }
        }
    }
    for(int i=0;i<cards.size();i++){
        cout<<cards[i]<<" ";
    }
    cout<<endl;
    return cards;
}
```

Function that generates a hand using randomCard function. Increments hand. Loops four times to get four different cards and until present cards are not generated. Gets random cards, checks conditions if card not found. Puts card in hand and breaks the loop.

```
void checkHand(vector<string> hand){
    vector<int> number;
    vector<char> suit;
    for(int i=0;i<4;i++){
        string number_string=hand[i].substr(0,hand[i].length()-1);
        number.push_back(atoi(number_string.c_str()));
        suit.push_back(hand[i][hand[i].length()-1]);
    }
```

checkHand function checks the type of hand. Extracts number and suit of the cards.

```
    sort(number.begin(), number.end());
    if(number[0]==number[1] && number[1]==number[2] && number[2]==number[3]){
        cout<<"Congrats: You got a Four of a kind and have won $400 for each chip"<<endl;
        wins++;
        chips+=400*bet;
```

Sorts the vector numbers and checks conditions for a four of a kind.

```
else if(suit[0]==suit[1] && suit[1]==suit[2] && suit[2]==suit[3]){
    if (number[0]==number[1]-1 && number[1]==number[2]-1 && number[2]==number[3]-1||number[0]==number[1]+1 && number[1]==number[2]+1 && number[2]==number[3]+1){
        cout<<"Congrats: You got a Straight Flush and have won $300 for each chip"<<endl;
        wins++;
        chips+=300*bet;
    }
```

If conditions don’t meet four of a kind, it moves to straight flush.

```
    else if(number[0]==number[1]-1 && number[1]==number[2]-1 && number[2]==number[3]-1||number[0]==number[1]+1 && number[1]==number[2]+1 && number[2]==number[3]+1){
        cout<<"Congrats: You got a Straight and have won $200 for each chip"<<endl;
        wins++;
        chips+=200*bet;
    }
```

If conditions don’t meet straight flush, it checks for a straight.

```
            else{
                cout<<"Congrats: You got a Flush and have won $250 for each chip"<<endl;
                wins++;
                chips+=250*bet;
            }
        }
```

If conditions meet for a flush but not for anything else, else statement prints out flush only.

```
        else if((number[0]==number[1] && number[1]==number[2]) ||
            (number[1]==number[2] && number[2]==number[3])){
            cout<<"Congrats: You got a Three of a kind and have won $150 for each chip"<<endl;
            wins++;
            chips+=bet*150;
        }
        else if(number[0]==number[1] && number[2]==number[3]){
            cout<<"Congrats: You got a Two Pairs and have won $100 for each chip"<<endl;
            wins++;
            chips+=100*bet;
        }
        else if((number[0]==number[1]) || (number[1]==number[2]) || (number[2]==number[3])){
            cout<<"Congrats: You got a Two of a kind and won $1 for each chip"<<endl;
            wins++;
            chips+=bet;
        }
        else{
            cout<<"Sorry: You got a Bubkiss and have lost $"<<bet<<endl;
            loss++;
            chips-=bet;
        }
    }
};
```

Checks if cards meet the conditions for three of a kind, two pairs, or two of a kind. If not, it becomes a bubkiss.

```
int main(){
    cout<<"Welcome to single player Poker game"<<endl;
    cout<<"Your initial bank roll is $100.00"<<endl<<endl;
    cout<<"================================================"<<endl;
    Poker poker;
    string y;
    while(1){
     cout<<"Play a hand [y / n]? ";
     cin>>y;
     if(y=="y" || y=="n"){
         break;
     }
     cout<<"Invalid input, should be y or n"<<endl;
    }
    while(y=="y"){
     cout<<endl<<"Place your bet [1, "<<poker.chips<<"]: ";
     cin>>poker.bet;
     if(poker.bet==0){
         cout<<poker.bet<<" Invalid input, cannot bet 0 chips"<<endl;
         continue;
     if(poker.bet>poker.chips){
         cout<<poker.bet<<" Invalid input, the input should be between 1 and "<<poker.chips<<endl;
         continue;

     }
     }
```

Prints out the welcome page and headers. Declares the poker object. Asks for input and checks the input. Asks for bet and checks if bet is valid, either in the bounds or not zero.

```
122         cout<<"... Shuffling Deck ..."<<endl;
123         cout<<"Let the cards fall where they may ..."<<endl;
124         vector<string> hand=poker.hand();
125         poker.checkHand(hand);
126         cout<<endl;
127         if(poker.chips<=0){
128             cout<<"Your balance is $0"<<endl;
129             break;
130         }
131         while(1){
132             cout<<"Play a hand [y / n]? ";
133             cin>>y;
134             if(y=="y" || y=="n"){
135                 break;
136             }
137             cout<<"Invalid input, should be y or n"<<endl;
138         }
139     }
```

Once the bet is valid, it starts to draw a hand for the player. Calls the hand and checks the hand. After that it checks the chip balance and asks for an input to play another hand and checks that input. If the chip balance has reached zero, the loop breaks and outputs the final results.

```
140     cout<<endl<<"=================================================="<<endl;
141     cout<<"Thanks for playing ..."<<endl;
142     cout<<"You playes a total of "<<poker.hands<<" hands"<<endl;
143     cout<<"Of which, you won "<<poker.wins<<endl;
144     cout<<"And you lost "<<poker.loss<<endl;
145     cout<<"But in the end you won $"<<poker.chips-100;
146 }
```

Once input has been given to not play another hand, it breaks the loop and displays the total of hands dealt, the total hands won, the total hands lost, and displays how much the player won or loss.

```
Play a hand [y / n]? y

Place your bet [1, 100]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
8S 7H 8H 7S
Congrats: You got a Two Pairs and have won $100 for each chip

Play a hand [y / n]?
```

Two pairs case.

```
Play a hand [y / n]? y

Place your bet [1, 1600]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
4S 5D 5H 6H
Congrats: You got a Two of a kind and won $1 for each chip
```

Two of a kind case.

```
Play a hand [y / n]? y

Place your bet [1, 1010]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
6D 5H 3S 4C
Congrats: You got a Straight and have won $200 for each chip

Play a hand [y / n]?
```

Straight case.

```
Play a hand [y / n]? y

Place your bet [1, 165]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
1C 4C 7C 3C
Congrats: You got a Flush and have won $250 for each chip

Play a hand [y / n]?
```

Flush case.

```
Play a hand [y / n]? y

Place your bet [1, 400]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
2C 1C 4C 3C
Congrats: You got a Straight Flush and have won $300 for each chip

Play a hand [y / n]?
```

Straight flush case.

```
Play a hand [y / n]? y

Place your bet [1, 900]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
2H 2C 9H 2S
Congrats: You got a Three of a kind and have won $150 for each chip

Play a hand [y / n]?
```

Three of a kind case.

```
Play a hand [y / n]? y

Place your bet [1, 550]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
5C 5D 5S 5H
Congrats: You got a Four of a kind and have won $400 for each chip

Play a hand [y / n]?
```

Four of a kind case.

```
Play a hand [y / n]? y

Place your bet [1, 200]: 5
... Shuffling Deck ...
Let the cards fall where they may ...
3S 6S 2C 5C
Sorry: You got a Bubkiss and have lost $5

Play a hand [y / n]?
```

Bubkiss case.

```
==================================================
Thanks for playing ...
You playes a total of 7 hands
Of which, you won 1
And you lost 6
But in the end you won $1220
```

Final Results.

```
==================================================
Play a hand [y / n]? y

Place your bet [1, 100]: 101
101 Invalid input, the input should be between 1 and 100
```

Invalid input

```
Place your bet [1, 100]: 100
... Shuffling Deck ...
Let the cards fall where they may ...
8C 2S 6S 9S
Sorry: You got a Bubkiss and have lost $100

Your balance is $0

==================================================
Thanks for playing ...
You playes a total of 1 hands
Of which, you won 0
And you lost 1
But in the end you won $-100
------------------------------
```

Balance becomes $0