Project 1

Title:
B.S.
(Bullshit)

Course:
CIS-5

Section Code:
40107

Due Date:
February 2, 2019

Author:
Ayah Seirafi

Table of Contents:

Introduction:

Using concepts from Chapters 1 - 5 in Starting out with C++ by Tony Gaddis, I was able to create a well known game called Bullshit. I used various libraries such as iostream, cmath, string, cstdlib, and ctime. Variables included bools, ints, chars and strings. Key concepts were implemented in the code as well. While only being able use 5 chapters as reference, I was unable to include concepts such as arrays and sorting. Because I was unable to use these, it was difficult to create a deck that would have been kept track of how many cards I was giving away and gaining. Rather that separating the cards into their suits, I simply gave them values one through thirteen.

Game Rules:

In the game Bullshit, the deck is split between all the players. In my game, there is only the live player, and the computer; therefore, each player would receive twenty-six cards. The goal of the game is to no longer have any card left in your hand. One player will lay an ace face up on the table. From there, the computer and the player will take turns laying down the next card in the order of: Ace, one, two, three, four, five, six, seven, eight, nine, ten, jack, king, and queen. Other than the ace, all other cards will be face down. If one player doesnt have the card that goes next, they will have to 'Bullshit' or give up and take the deck. If the player or the computer suspect that the other player is bullshitting and is not putting the correct card down, they can call bullshit on the other player. If the player that calls bullshit is correct, and the other

player faked their card, then the player that faked the card will have to take the whole pile. This will continue until one player gets rid of all the cards in their hand.

Game Setup:

In my program, I created two function prototypes. I had one called *grabRandomCard()* and *runGame()*. In *grabRandomCard()*, I created a switch statement that layed out the value of each card. I then had an integer titled 'num'. This number would be randomized and given to the player to place down. The bulk of the game would be run in the *runGame()* prototype. I first asked the player to enter their name before the game began to make it more interactive. I then tell the user to hit enter to pick a card. They are shown this card, then given the option to B.S. or just not risk it and take the whole pile. If they have the correct card, they will only be asked if they want to give the card away. After the player puts their card down, Player 2 (which is the computer) will call out B.S. only twenty-five percent of the time. When it will be called is random. After, player two will auto

matically put their card down, and player 1 will be given the opportunity to call B.S. on the computer. This will go on until each players cards have decremented to zero. Then a player will be declared winner. At the end, player one will be asked if they would like to play another round. If they answer either 'y' or 'n' in capital or lowercase letters, the game will either start over or end.

Program:

```
/*
 * File:   main.cpp
 * Author: Ayah Seirafi
 *
 * Created on February 2, 2019, 10:19 PM
 */

#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>
using namespace std;

//prototypes
int grabRandomCard();
void runGame();

//execution begins here
int main()
{
    runGame(); // prototype to run game
```

```
    char choice; // choice for player to say yes or no


    // ask user if they want to after the game was run
    cout << "Do you want to play again? ";
    cin >> choice;
    //if they say yes, we will run the game again
    while(choice == 'Y' || choice == 'y')
    {
        runGame();
        cout << "Do you want to play again? ";
        cin >> choice;
    }
    return 0;
}
//function
void runGame()
{
    //Declare Variables
    int plr1cnt = 26; // player one will start with 26 cards
    int plr2cnt = 26; // player two will start with 26 cards
    int pile = 1; // the card pile will start with 1 card
    int face = 1; // the pile will start with 1 (which is ace)
```

```
    int bsProb = 0; // probability for computer to call BS
    int reqCard = 0; // next card needed to be placed on pile
    char option;  // users option to call BS
    bool isGamOv = false; // does either player get to 0 cards in thir game
    bool match = true; // do the card in their hands hatch what they need
    string name; // player 1 name

    // random seed generator
    srand(time(NULL));

   // game starts here
    cout << "Welcome to BS!  Please enter your name. " << endl;
    cin >> name;
    cout << "Okay, good luck " << name << "! Lets see how well you can BS a computer!" <<
endl;
    cout << "Game started! the current face card is: " << face << endl;

    do
    {
        //player one
        cout << "\nPLAYER 1'S TURN:";
        cout << "\nPress the Enter key to place down the required card: ";
```

```
getchar();  // get card
cin.ignore();
reqCard = grabRandomCard();
pile++; // the pile will increase when player places card down
cout << "you have a " << reqCard << endl; // state hand


if(reqCard == (face + 1)) // next card
{
    // give user option to place correct card down
    cout << "do you want to place down the card? [Y/N] ";
}
else
{
    // give user option to BS
    cout << "do you want to BS? [Y/N] ";
}
// have user input whether they want to BS
cin >> option;
if(option == 'Y' || option == 'y') // yes they want to BS
{
    plr1cnt--; // player 1 loses card after one is laced down
```

```
if(face == 13) // this will start deck over again
{
    face = 0;
}
if(reqCard == (face + 1)) // next card
{
    match = true; // card matches
}
else
{
    match = false; // card doesn't match
}

//25% chance plyr 2 will call BS
bsProb = rand() % 4 + 1;
if(bsProb == 4)
{
    cout << "Player 2 called BS!" << endl;
    if(match == false) // player that we called BS on was correct
    {
        cout << "Player 2 is correct! You picked up the pile"
        << "\n";
```

```
            plr1cnt += pile; // player 1 takes the pile
        }
        else
        {
            // if BS is called and it was wrong
            cout << "Sorry, Player 1 placed down the correct card\n";
            cout << "Player 2 picked up the pile" << endl;
            plr2cnt += pile; //player 2 takes pile
        }
    }

    if(plr1cnt == 0) // player 1 runs out of cards
    {
        cout << "Player 1 Wins!" << endl;
        isGamOv = true;
    }
}
else
{
    cout << "You picked up the pile" << endl;
    plr1cnt += pile;
}
```

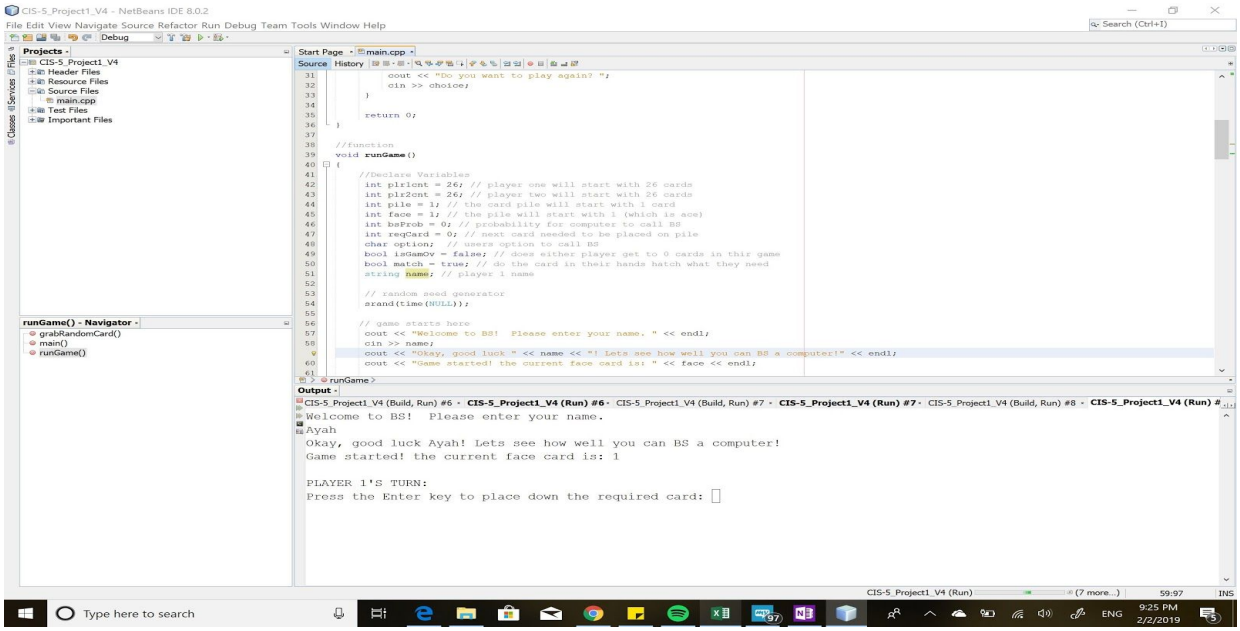
```
//player two
if(isGamOv == false)
{
    cout << "\nPLAYER 2'S TURN:" << endl;
    cout << "Player 2 placed down a card" << endl;
    reqCard = grabRandomCard();

    plr2cnt--; // player 2 loses card

    if(face == 13) // start over when card reaches 13
    {
        face = 0;
    }
    if(reqCard == (face + 1)) // next card
    {
        match = true; // card matched
    }
    else
    {
        match = false; // card doesn't match
```

```
}
// give player 1 option to call BS
cout << "Do you want to call BS on Player 2? [Y/N] ";
cin >> option;
if(option == 'Y' || option == 'y')
{
    if(match == false) // cals BS
    {
        cout << "Player 1 is correct! Player 2 picked up the pile" << endl;
        plr2cnt += pile; // player 2 must take pile
    }
    else // if player 2 places down correct card
    {
        cout << "Sorry, Player 2 placed down the correct card\n";
        cout << "You picked up the pile" << endl;
        plr1cnt += pile; // player 1 takes pile
    }
}

if(plr2cnt == 0) // when player 2 runs out of cards, player 2 wins
{
    cout << "Player 2 Wins!" << endl;
```

```
            isGamOv = true;
        }
    }

    }while(isGamOv == false); // game over
}




//function to generate random card
int grabRandomCard()
{
    int card = 0;
    int num = rand() % 13 + 1;
    switch(num)
    {
        case 1: card = 1; break;
        case 2: card = 2; break;
        case 3: card = 3; break;
        case 4: card = 4; break;
        case 5: card = 5; break;
        case 6: card = 6; break;
```

```
        case 7: card = 7; break;

        case 8: card = 8; break;

        case 9: card = 9; break;

        case 10: card = 10; break;

        case 11: card = 11; break;

        case 12: card = 12; break;

        case 13: card = 13; break;

    }


    return card;

}
```

Output Photos and Explanations:



In this first photo, the user is asked to input their name. The face card (which is 1) is what is considered to be the Ace of the

```
9   #include <cstdlib>
runGame > do > if(isGamOv == false) > then >
Output
CIS-5_Project1_V4 (Build, Run)   CIS-5_Project1_V4 (Run)
Game started! the current face card is: 1

PLAYER 1'S TURN:
Press the Enter key to place down the required card:

you have a 6
do you want to BS? [Y/N] y

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N] n
```

In this photo, the user is prompted to hit ‘enter’ to know the card that they will have to put down. They are then given the option to bullshit and put the card down if it isn't the right one, or they can just pick up the pile if they do not want to bullshit. The computer then puts down its card and the user is given the opportunity to callbullshit on the computer.

```
9   #include <cstdlib>
runGame > do > if(isGamOv == false) > then >
Output
CIS-5_Project1_V4 (Build, Run)   CIS-5_Project1_V4 (Run)
Game started! the current face card is: 1

PLAYER 1'S TURN:
Press the Enter key to place down the required card:

you have a 6
do you want to BS? [Y/N] y

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N] n

PLAYER 1'S TURN:
Press the Enter key to place down the required card:
you have a 2
do you want to place down the card? [Y/N]
y
Player 2 called BS!
Sorry, Player 1 placed down the correct card
Player 2 picked up the pile

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N]
```

```
do you want to BS? [Y/N] y

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N] y
Player 1 is correct! Player 2 picked up the pile

PLAYER 1'S TURN:
Press the Enter key to place down the required card:
you have a 3
do you want to BS? [Y/N] y

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N] y
Player 1 is correct! Player 2 picked up the pile

PLAYER 1'S TURN:
```

This is where the user and computer call B.S. on each other

```
Output
CIS-5_Project1_V4 (Build, Run) #4 | CIS-5_Project1_V4 (Run) #4 | CIS-5_Project1_V4 (Build, Run) #5 | CIS-5_Project1_V4 (Run) #5 | CIS-5_Project1_
Game started! the current face card is: 1

PLAYER 1'S TURN:
Press the Enter key to place down the required card:

you have a 6
do you want to BS? [Y/N] y
Player 1 Wins!
Do you want to play again? n
read from master failed
                        : Input/output error

RUN FAILED (exit value 1, total time: 7s)
```

```
168                 }
169
```

runGame > do > if(isGamOv == false) > then > if(plr2cnt == 25) >

Output

CIS-5_Project1_V4 (Build, Run) #4 | **CIS-5_Project1_V4 (Run) #4** | CIS-5_Project1_V4 (Build, Run)

```
Game started! the current face card is: 1

PLAYER 1'S TURN:
Press the Enter key to place down the required card:

you have a 4
do you want to BS? [Y/N] n
You picked up the pile

PLAYER 2'S TURN:
Player 2 placed down a card
Do you want to call BS on Player 2? [Y/N] n
Player 2 Wins!
Do you want to play again? y
Game started! the current face card is: 1

PLAYER 1'S TURN:
Press the Enter key to place down the required card:
```

This is when the game was set to only needing 25 cards in your hand to win. The computer won this one easily because I decided not to bullshit. The computer put their card down and immediately won.