**PROJECT 2**

**[Blackjack Simulation]**

[**19SUM-CSC-5-45779**](https://rccd.instructure.com/courses/14792)

**Professor Mark Lehr**

**Shravan Shenoy**

**GITHUB REPOSITORY LINK:** [**https://github.com/shravanrcc123/CSC5Project2Final**](https://github.com/shravanrcc123/CSC5Project2Final)

**PROJECT 2 CHECKLIST LINK (Github):**

[**https://github.com/shravanrcc123/CSC5Project2Final/tree/master/Project%202%20Checklist**](https://github.com/shravanrcc123/CSC5Project2Final/tree/master/Project%202%20Checklist)

**Introduction**

Blackjack, also known as ‘21’, is a classic card game, one that, with the right functions and knowledge of programming, can be seamlessly integrated and executed in the C++ language. The objective of the game is for the player to beat the dealer by scoring as close to 21, without going over it, with the cards that have been dealt by him or her. Cards are shuffled and dealt by the dealer, who has his own set of cards. Players can ‘hit’, meaning be dealt another card and take their chances coming closer to 21 without going over. Or, they can ‘stand’ and let the dealer draw as many cards as the player had hit beforehand. The values are then totaled up after the player stands and the dealer draws cards accordingly, and the winner is whoever is NOT over 21 but is closest to it. An important note is that King, Queen, and Jack cards are worth 10, with an Ace being special. An Ace can either be valued at 1, or 11. This is dependent on whether or not the player will go above 21 if they are dealt an Ace. For example, if a player already has a total card value of 16, and is then dealt an Ace, that Ace will be worth 1 and not 11 so as to not go over 21. Players can bet money on the game as well.

**Understanding the Game – A Typical Round**

A typical round of Blackjack is as follows:

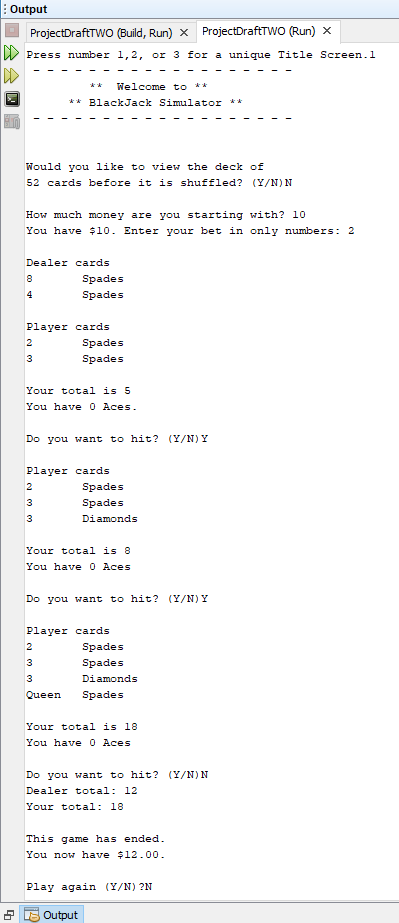
The dealer shuffles a deck of 52 cards and deals 2 cards to the player and himself.

The player is able to look at both of his/her cards, as well as the dealers. However, the dealers cards may only be viewed in the first round and none thereafter unless the player chooses to ‘stand’.

The player can hit as many times as they want so long as he/she does not surpass 21.

Once the player stands, the dealer will then draw cards and the winner will be revealed.

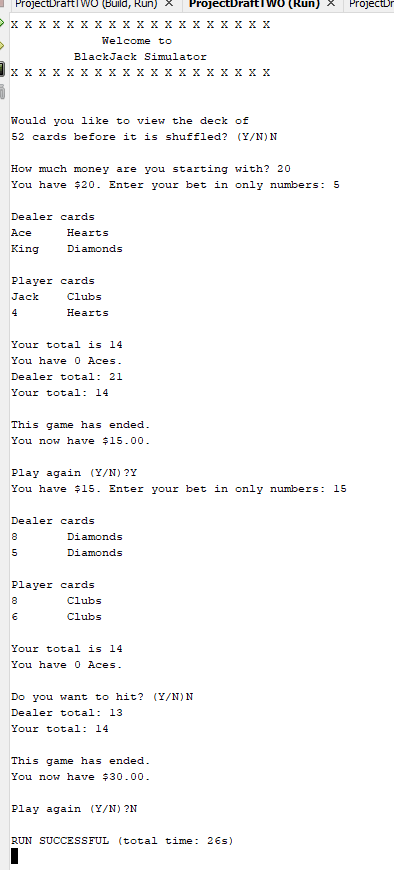
**Sample Trial Run Images**

In the event that the raw code at the latter end of the document is unable to work, I have provided some screenshots of trials that were executed with near final/final versions of the code.

**IMAGE 1**

This image shows a trial of the Blackjack Simulator, with Custom Menu option 1, and with the player choosing to NOT view the 2D Array of the card deck before it is shuffled.

A single game is played and displayed here before the run ends.

** PLEASE SEE NEXT PAGE**

**IMAGE 3**

**PSEUDOCODE RUNDOWN**

1 round is played here.

Heres some pseudocode:

Switch case: pick 1,2,3 to display a unique menu

Would you like to view this 2D array of the deck? (Y/N)

If yes, display 2D array, if no, move on

How much money are you starting with? This will be your total money for now

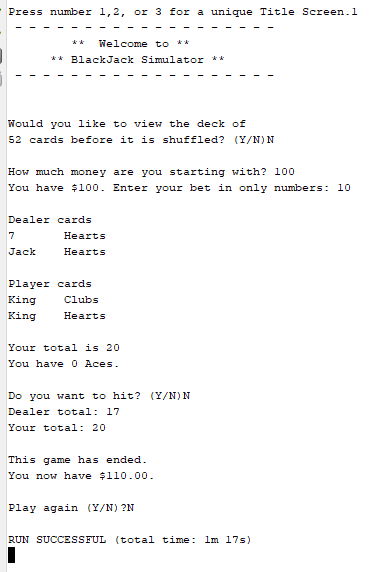
How much money do you want to bet? This is the amount you will either win or lose, so this will be either added or subtracted from the total.

Randomized Deck will be dealt, showing what Face it is as well as the Value, unless it is a King, Queen, Jack or Ace

The program then displays your total, your number of Aces, as well as the dealers current total (the dealers total is shown only in the first round.

Checks whether or not anyone has won/lost yet by scoring 21 or scoring above 21.

Yes, both the player can win within the first round!



**IMAGE 3**

**PSEUDOCODE RUNDOWN**

1 round is played here.

Heres some pseudocode:

Switch case: pick 1,2,3 to display a unique menu

Would you like to view this 2D array of the deck? (Y/N)

If yes, display 2D array, if no, move on

How much money are you starting with? This will be your total money for now

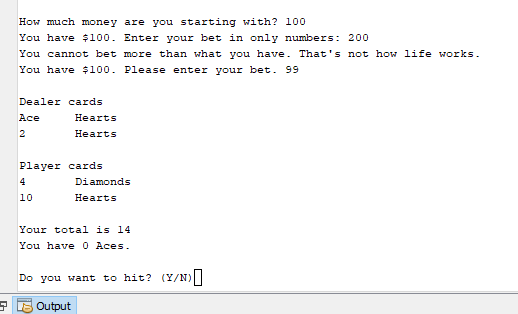
How much money do you want to bet? This is the amount you will either win or lose, so this will be either added or subtracted from the total.

Randomized Deck will be dealt, showing what Face it is as well as the Value, unless it is a King, Queen, Jack or Ace

The program then displays your total, your number of Aces, as well as the dealers current total (the dealers total is shown only in the first round.

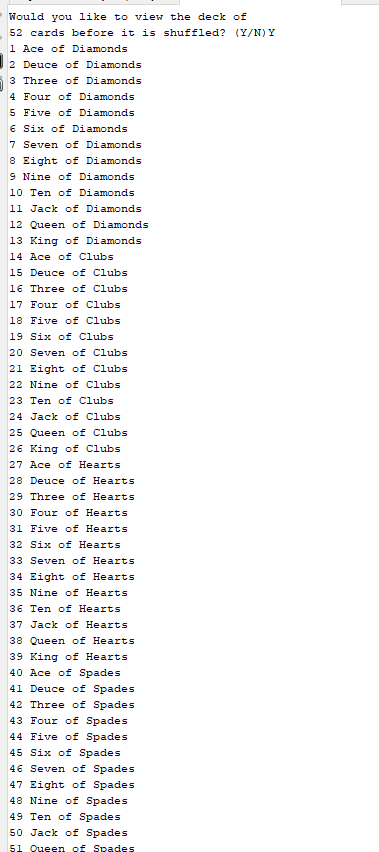
Checks whether or not anyone has won/lost yet by scoring 21 or scoring above 21.

Yes, both the player can win within the first round!



**IMAGE 4**

Displays what unique output display happens if the player tries betting more than the amount of money they have.



**IMAGE 5**

This image shows the 2D Array of cards that the player has the option of viewing before officially starting the game and having the dealer shuffle the deck.

**RAW PROGRAM CODE:**

/\*

\* File: main.cpp

\* Author: Shravan Shenoy

\* Created on July 21, 2016, 2:15 PM

\* Purpose: Project 2 Blackjack Simulation

\*/

//System Libraries

#include <iostream> // Input/Output Stream Library

#include <iomanip> // Formatting Library

#include <ctime> // Unique Seed Value Library

#include <cstdlib> // Random Value Library

#include <string> // String Library

#include <fstream> // File I/O

#include <cmath> // Math Library

using namespace std;

void deal(int,int&,int& ,int[][2],int&,int[][2] ); // function overload

void deal(int,int&,int&,int&,int[][2],int&,int[][2] ); // function overload

int getbet(int); //returns int

void fillDc(int[][2],int); // populates deck to be shuffled

void SHUFFL(int[][2],int); // uses rand() to ensure randomness in shuffling

void couthnd(int,int[][2]); // shows player hand

int main(int argc, char\*\* argv) {

const float PERCENT=100.0f;//Conversion to Percent

const int HNDRDMIL=10e8; //Hundred Million

const int MILLION =10e6; //Million

const int HNDTHSND=10e5; //Hundred Thousand

{

string suit[4]={"Hearts","Diamonds","Spades","Clubs"}; // setup array of 1st set of card catagory

string card[13]={"Ace","2","3","4","5","6","7","8","9","10","Jack","Queen","King"}; // setup array of 2nd set of card catagory

int ttlD; // total num of card values for Dealer

int ttlP; // total num of cards values for Player

static int STATICZ=0; // zero value to be changed

float money; // number of money player says he/she has at start

int bet; // amount of money used as bet

int aceNum; // num of aces

int ttlDECK=2; // number of decks (only 2, for player and dealer), is initialized and manipulated

int j; // used for deck functions

int FULDECK[52\*ttlDECK][2];

int dealhnd[20][2]; // showing dealers hand setup

int playhnd[20][2]; // showing players hand setup

int playcrd; // play card setup

int dealcrd; // deal card setup

char choice1='Y'; // Universal input choice for hitting and standing, and if player wants to play again

bool again; // bool option to determine if player wants to play again or not

srand(static\_cast<unsigned int>(time(0)));

ofstream out;

//Open File & Enter Primary Input Data

out.open("stats.dat");

int ttlscrn;

cout << "Press number 1,2, or 3 for a unique Title Screen."; // SWITCH CASE for title screen options

cin >> ttlscrn;

switch (ttlscrn)

{

case 1: cout<<" - - - - - - - - - - - - - - - - - - -"<< endl; // Unique Menu Option 1

cout<<" \*\* Welcome to \*\*" << endl;

cout<<" \*\* BlackJack Simulator \*\*" << endl;

cout<<" - - - - - - - - - - - - - - - - - - -\n"<< endl;

cout<< endl;

break;

case 2: cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<< endl; // Unique Menu Option 2

cout<<" Welcome to " << endl;

cout<<" BlackJack Simulator" << endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\n"<< endl;

cout<< endl;

break;

case 3: cout<<"X X X X X X X X X X X X X X X X X X X "<< endl; // Unique Menu Option 3

cout<<" Welcome to " << endl;

cout<<" BlackJack Simulator" << endl;

cout<<"X X X X X X X X X X X X X X X X X X X \n"<< endl;

cout<< endl;

break;

}

char choice2;

cout<<"Would you like to view the deck of \n52 cards before it is shuffled? (Y/N)";

cin>> choice2;

if (choice2 =='Y') // Choice 2, different from the while loop for later choice

{

const int rows2D = 4;

const int cols2D = 13;

string spclnme[rows2D] = {"Diamonds", "Clubs", "Hearts", "Spades"}; // setup array of 1st set of card catagory of 2D array

string crdFACE[cols2D] = {"Ace", "Deuce", "Three", "Four", "Five", "Six",// setup array of 2nd set of card catagory of 2D array

"Seven", "Eight", "Nine", "Ten", "Jack", "Queen", "King"};

int deck2D[rows2D][cols2D] = // Final setup of 2D array. Used for numbering unique cards

{

{1,2,3,4,5,6,7,8,9,10,11,12,13},

{14,15,16,17,18,19,20,21,22,23,24,25,26},

{27,28,29,30,31,32,33,34,35,36,37,38,39},

{40,41,42,43,44,45,46,47,48,49,50,51,52}

};

for(int i = 0; i < rows2D; i++) // For loop for 2D card array display

{

for(int j = 0; j < cols2D; j++)

{

cout << deck2D[i][j] << " " << crdFACE[j] << " of " << spclnme[i] << endl;

}

}

}

fillDc(FULDECK,ttlDECK); // fills the deck with cards, initializes

SHUFFL(FULDECK,ttlDECK); // shuffles the deck, ensures randomness

cout<<"\nHow much money are you starting with? ";

cin>>money;

while(money>0&&toupper(choice1)=='Y') // Conditions before using exit()

{

bet=getbet(money); // keeps bet equal to money accurately through return int function

playcrd=0; // initial value for players cards, used in cout hand function

dealcrd=0; // initial value for dealers cards

cout<<"\nDealer cards\n";

ttlD=0; // total points for dealer

deal(2,ttlD,STATICZ,FULDECK,dealcrd,dealhnd);

couthnd(dealcrd,dealhnd);

aceNum=0; // initial number of aces in players hand

ttlP=0; // total points foe player

cout<<"\nPlayer cards\n";

deal(2,aceNum,ttlP,STATICZ,FULDECK,playcrd,playhnd);

couthnd(playcrd,playhnd);

cout<<"\nYour total is "<<ttlP<<endl;

cout<<"You have "<<aceNum<<" Aces."<<endl;

again=false;

while(ttlD<21&&ttlP<21&&!again) // Checks win/lose condition to hit again

{

cout<<"\nDo you want to hit? (Y/N)"; // option to hit (Y/N)

cin>>choice1;

if(toupper(choice1)=='Y') // Essentially, continues running the game

{cout<<"\nPlayer cards\n";

deal(1,aceNum,ttlP,STATICZ,FULDECK,playcrd,playhnd);

couthnd(playcrd,playhnd);

cout<<"\nYour total is "<<ttlP<<endl;

cout<<"You have "<<aceNum<<" Aces"<<endl;

}

else

again=true;

}

cout<<"Dealer total: "<<ttlD<<endl; // total card values for Dealer

cout<<"Your total: "<<ttlP<<endl; // total card values for Player

if(ttlD>ttlP||ttlP>21) // If player loses in last round, money is lost

money-=bet;

else if(ttlD<ttlP&&ttlP<=21) // If the player wins in last round, money is won

money+=bet;

cout<<fixed<<setprecision(2)<<showpoint; // formatting of resulting money from game

cout<<"\nThis game has ended. \nYou now have $"<<money<<".\n"; // Formatted in $ .00 style

if(money>0) // As long as the players money is above 0, they can play again.

{cout<<"\nPlay again (Y/N)?";

cin>>choice1;

}

//Finishing Outputs to file

out << "Here are your Final Game Stats:" << endl << endl;

out << "You ended with a total of " << money <<".";

out.close();

}

exit(0); // Ends run after player decides to say 'N' when asked if he/she wants

// to play another game

// OR exits when player runs out of money

return 0;

}

}

void couthnd(int m,int FULDECK[][2]) // shows card

{int i;

string suit[4]={"Hearts","Diamonds","Spades","Clubs"}; // first array for card values

string card[13]={"Ace","2","3","4","5","6","7","8","9","10","Jack","Queen","King"}; // second array for card values

for(int i=0;i<m;i++)

cout<<card[FULDECK[i][1]]<<"\t"<<suit[FULDECK[i][0]]<<endl;

}

void deal(int number,int& TTLVAR,int& STATICZ,int FULDECK[][2],int& m,int p[][2]) // VERSION 1 OVERLOAD function that deals actual cards using static variable

{int i,n;

for(i=0;i<number;i++)

{n=FULDECK[STATICZ][1];

n++;

if(n>10)

n=10;

p[m][1]=FULDECK[STATICZ][1];

p[m][0]=FULDECK[STATICZ][0];

m++;

if(n==1)

TTLVAR+=11;

else if(n>=10)

TTLVAR+=10;

else

TTLVAR+=n;

STATICZ++;

}

}

void deal(int number,int& aceNum,int &TTLVAR,int& STATICZ,int FULDECK[][2],int& m,int p[][2]) // VERSION 2 OVERLOAD function that deals actual cards using static variable

{int i,n;

for(i=0;i<number;i++)

{n=FULDECK[STATICZ][1];

n++;

if(n>10)

n=10;

p[m][1]=FULDECK[STATICZ][1];

p[m][0]=FULDECK[STATICZ][0];

m++;

if(n==1)

{aceNum++;

TTLVAR+=11;

}

else if(n>=10)

TTLVAR+=10;

else

TTLVAR+=n;

STATICZ++;

}

if(TTLVAR>21)

if(aceNum==0)

return;

else

{aceNum--;

TTLVAR-=10;

}

}

int getbet(int money) // function that returns int, displays money and gets user input for bet

{int bet;

cout<<"You have $"<<money<<". Enter your bet in only numbers: ";

cin>>bet;

while(bet>money)

{cout<<"You cannot bet more than what you have. That's not how life works.\n";

cout<<"You have $"<<money<<". Please enter your bet. ";

cin>>bet;

}

return bet;

}

void SHUFFL(int FULDECK[][2],int ttlDECK) // function that uses rand() to shuffle deck

{

int q,num1,type1,num2,type2,TEMPVAL;

for(q=0;q<100\*ttlDECK;q++)

{num1=rand()%(13\*ttlDECK);

num2=rand()%(13\*ttlDECK);

type1=rand()%4;

type2=rand()%4;

TEMPVAL=FULDECK[type1][0];

FULDECK[type1][0]=FULDECK[type2][0];

FULDECK[type2][0]=TEMPVAL;

TEMPVAL=FULDECK[num1][1];

FULDECK[num1][1]=FULDECK[num2][1];

FULDECK[num2][1]=TEMPVAL;

}

}

void fillDc(int FULDECK[][2],int ttlDECK=2) // function that populates the deck of cards

{

bool CRDBOOL[4][13][ttlDECK];

int a,b,c,num,type,d;

for(a=0;a<4;a++)

for(b=0;b<13;b++)

for(c=0;c<ttlDECK;c++)

CRDBOOL[a][b][c]=false;

for(b=0;b<52\*ttlDECK;b++)

{do

{

num=rand()%13;

type=rand()%4;

d=rand()%ttlDECK;

}while(CRDBOOL[type][num][d]);

FULDECK[b][0]=type;

FULDECK[b][1]=num;

CRDBOOL[type][num][d]=true;

}

}

/\*

void Arryfll(int array[], int SIZE) {

for (int i = 0; i < SIZE; i++) {

"11 18 21 22 24 24 24 24 28 29 \n" <<

"29 31 33 33 36 40 41 44 48 50 \n" <<

"51 53 55 55 55 57 57 58 61 64 \n" <<

"65 67 68 68 72 72 74 74 80 80 \n" <<

"81 81 83 83 83 87 87 88 90 96 \n" <<

"99 99\n" >> array[i];

}

}

void Sortsel(int array[], int SIZE) {

int temp;

for (int i = 0; i < SIZE; i++) {

for (int j = i + 1; j < SIZE; j++) {

if (array[i] > array[j]) {

temp = array[i];

array[i] = array[j];

array[j] = temp;

}

}

}

}

void print(int array[], int SIZE) {

for (int i = 0; i < SIZE; i++) {

cout << array[i] << " ";

if (i == 9 || i == 19 || i == 29 || i == 39 || i == 49 || i == 59 || i == 69 || i == 79 || i == 89) {

cout << endl;

}

}

cout << endl << endl;

}

bool Searchbn(int[], int, int, int & );

const int SIZE=52;

int array[SIZE];

int indx,val;

Arryfll(array,SIZE);

Sortsel(array,SIZE);

print(array,SIZE);

cout << "Input the value to find in the array" << endl;

cin>>val;

for (int i=0; i<SIZE; i++){

if (array[i] == val){

cout << val << " was found at indx = " << i << endl;

return 0;

}

}

void prtoppN(string oppname1)

{

cout << "Blackjack Bot";

}

void prtoppN(string oppname2)

{

cout << "CSC Student";

}

void prtoppN(string oppname3)

{

cout << "Bob";

}

void OppSlct(string oppname)

{

string oppname='A';

cout<<"What do you want the name of your opponent to be?"<<endl;

cout<<"Press A and enter for ' CSC Student ',\n or anything else and enter for ' Blackjack Bot '.\n";

cin>>oppname;

(oppname=='A') ? (oppname='Student') : (oppname='Bot');

cout << "The name of your opponent is '" << oppname << "'.\nLet's begin the game.";

string oppname,oppname1,oppname2,oppname3;

cout<<"What do you want the name of your opponent to be?"<<endl;

cout<<"Press A and enter for ' Blackjack Bot ',\nB for ' CSC Student ',\nor C for 'Bob'.\n";

cin>>oppname;

cout << "The name of your opponent is:\n";

if (oppname=='A')

{

prtoppN("Blackjack Bot");

}

if (oppname=='B')

{

prtoppN("CSC Student");

}

if (oppname=='B')

{

prtoppN("Bob");

}

}

\*/