Project 2

<Double or Nothing

A Game By Bradley McKenzie>

Course: CSC-5 Section: 40107

Due Date: 02/09/17

Author: Bradley McKenzie

**Introduction**:

Title: Double or Nothing

First you must start by inputting a username and your password (**password=password**). If password is incorrect the player will be prompted to try again. This is a betting game in which, a player in puts a bet. Followed by a color to bet on (Red or Blue). If the player gets the bet right then they bet will be doubled. However, if they get the bet wrong then they lose the amount they put down. The program will tell the player their current winnings.

For example: I put a bet down of $1000 on red.

The spinner landed on: Red

The payout is: $2000

Current Total: $2000

For example: I put a bet down of $1000 on red.

The spinner landed on: Blue

The payout is: $-1000

Current Total: $1000

The percentage of wins and losses are displayed after all games are played. The possible winnings are displayed at the end of the number of games the user plays, as well as the total winnings earn or loss in that game.

For example:

-----------------------------

Percentage Wins of = 50%

Percentage of Losses = 50%

Possible Winnings: $4000

Total Winning: $1000

-----------------------------

Nice Job!

Thank You, For Playing.

A Game By Bradley McKenzie

**Summary**:

Number of Lines: 272

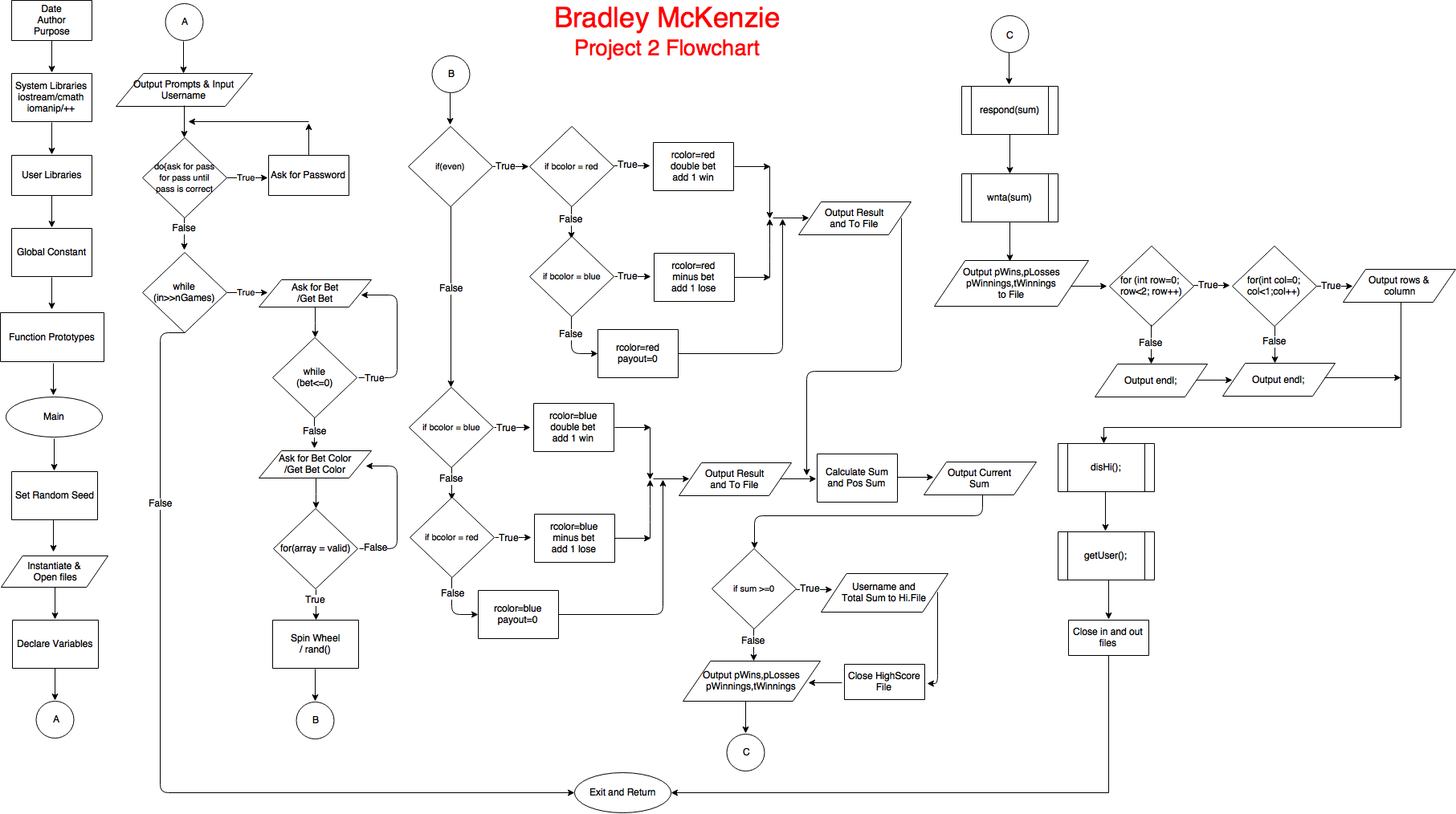
Number Lines of Comments: 112

**Description**:

The standard number of games is 3; however the number of games can beset with the file named: "NumberOfGames.dat". Each result per game will be displayed and outputted to a file named: "ResultsOfGames.dat". If the player enters an invalid bet color, the program will tell them it was an invalid bet and that game will not count against their total parentage wins and losses. They player can enter "Red", "red", "RED", and "r", same goes for blue as well, for their bet and the program will still process the game result. If they player entered again else beside what was state before then the program will tell they player "Invalid: Bet Color Must be (Red or Blue)" If the player made money in the game, that result will be put into the "HighScoreOfGame.dat" file along with other games played before high scores.

The player will see responses base on how well they did in that game. If the player makes greater than zero and less than five million then the program will respond with "Nice Job". If the player makes 5 million or more the program will respond with "Quitting Your Job?", and last if player losses money it will respond with "Better Luck Next Time" and "Why Not Try Again". You are able to search for a username at the end of each game, all you have to do is enter "Y" when prompted, if you wish to see the high scores enter "N".

**Flowchart:**



**Pseudo Code**:

/\*

File: main.cpp

Author: Bradley McKenzie

Created on February 6, 2017

Purpose: Project 2

\*/

//System Libraries

//Standard library

//Random Numbers/Exit

//Time to set the seed

//File stream library

//Format library

//String library

//Vector library

//User Libraries

//Global Constants

//Such as PI, Vc, -> Math/Science Values

//as well as conversions from system of units to another

//Calculate Wins and Losses Percentages

//Limit Number of Games 100's

//Limit Number of Games 10's

//2D Array

//Function Prototypes

//Executable code begins here!!!

//Set the random number seed

//Instantiate and Open files

//Input Number of Games

//Output Result of Games

//Output Score if Winnings > 0

//Declare Variables

//Bet Amount

//Pay Out Result

//Total Winnings

//Possible Wins

//Number of Wins

//Number of Losses

//Password

//Username

//Bet Color

//Result Color

//Initialize the 2-D Array

//Input Values

//Get Username

//Get Correct Password

//If password is wrong ask for it again

//Loop to end

//Limit the number of games

// Loop the Game

//Bet amount input

//Not allow bet to negative or 0

//Bet amount input

//Input validation loop

//Bet color input

//Check if bet color is valid

//if !valid, then output invalid message

//Bet color input

//end input validation loop

//Process by mapping inputs to outputs

//Last line will be number of games from file

//Call random number generator for the color landed on

//Value from 1 to 26 //Spin Wheel

//Land on Red

//Bet Won

//Double Bet

//Add plus one win

//Bet Loss

//Minus Bet

//Add plus one loss

//Invalid Input

//Output Game Result for Land on Red

//Land on Blue

//Bet Won

//Double bet

//Add plus one win

//Bet Loss

//Invalid Input

//Output Game Result For Land on Blue

//Get Sum of Pay Out

//Get Sum of Bet

//Display Current Winnings

//Output Username and Total Winnings if Winnings is > 0

//Close High Score File

//Output Percentage of Wins and Losses

//Percent Wins

//Percent Losses

//Possible Winnings

//Actual Winnings

//Output Result Based on Winnings

//Percent Wins

//Percent Losses

//Possible Winnings

//Actual Winnings

//Thank player for playing the game

//Output rows and column

//Display High Score From File

//Get Y or N input then show high scores or exit

//Find a username in high score file

//Close Files and Exit stage right!

//Calculate Wins and Loss Percentage

//Output a respond based on Winnings

//Loss money respond

//Won money respond

//Won a lot of money respond

//Respond if player losses money

//Display High Score Yes or No

//Declare option to display high score

//Get Yes or No

//Input Y or y to open high score

//Input each line of the file

//Search for a username

//Input High Score File

**Cross Reference**:

Where in Code

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter** | **Section** | **Topic** | **Linenumber** |
| 2 | 2 | cout | Everywhere |
|  | 3 | libraries | iostream,iomanip,cmath,cstdlib,fstream,++ string, ctime |
|  | 4 | variables/literals | 46 |
|  | 5 | Identifiers | 46 |
|  | 6 | Integers | 46 |
|  | 7 | Characters | 46 |
|  | 8 | Strings | 46 |
|  | 9 | Floats NoDoubles | No Doubles |
|  | 10 | Bools | 93,219 |
|  | 11 | Sizeof\*\*\*\*\* |  |
|  | 12 | Variables 7 characters orless | 46 |
|  | 13 | Scope \*\*\*\*\* No GlobalVariables |  |
|  | 14 | Arithmeticoperators | 112,136,214 |
|  | 15 | Comments20%+ | Everywhere |
|  | 16 | NamedConstants | 23-25 |
|  | 17 | Programming Style \*\*\*\*\*Emulate | Everywhere |
|  |  |  |  |
| 3 | 1 | cin | Everywhere |
|  | 2 | MathExpression | 112,136,214 |
|  | 3 | Mixing data types\*\*\*\* |  |
|  | 4 | Overflow/Underflow\*\*\*\* |  |
|  | 5 | TypeCasting | 37 |
|  | 6 | Multiple assignment\*\*\*\*\* |  |
|  | 7 | Formattingoutput | Everywhere |
|  | 8 | Strings | 46 |
|  | 9 | MathLibrary | 9 |
|  | 10 | Hand tracing \*\*\*\*\*\* |  |
|  |  |  |  |
| 4 | 1 | RelationalOperators | 85,95,++ |
|  | 2 | if | 95,99,109++ |
|  | 4 | If-else | 121 |
|  | 5 | Nesting | 82 |
|  | 7 | Flags\*\*\*\*\* |  |
|  | 8 | Logicaloperators | 77,109,++ |
|  | 11 | Validating userinput | 85,94,++ |
|  | 13 | ConditionalOperator | 80 |
|  | 14 | Switch |  |
|  |  |  |  |
| 5 | 1 | Increment/Decrement | 113,119,++ |
|  | 2 | While | 79,85 |
|  | 5 | Do-while | 72 |
|  | 6 | Forloop | 82,94,++ |
|  | 11 | Files input/outputboth | 40-45,226++ |
|  | 12 | No breaks in loops\*\*\*\*\*\* |  |
|  |  |  |  |
| 6 | 3 | Prototypes | 27 |
|  | 4 | Pass byValue | 53,60,++ |
|  | 8 | ReturningValues | 204 |
|  | 9 | Boolean returnvalues | 219,266 |
|  | 11 | Static LocalVariable | 49,50,++ |
|  | 12 | DefaultArguments | 30 |
|  | 13 | ReferenceVariables |  |
|  | 14 | Overloadingfunctions |  |
|  |  |  |  |
| 7 | 4 | ArrayInitialization | 60-63 |
|  | 7 | ParallelArrays | 61,62 |
|  | 8 | Arrays in functionarguments | 32 |
|  | 9 | 2 DimensionalArrays | 63,184-189 |
|  |  | 2 Darrays as functionargument | 26 |
|  | 12 | STLVector |  |
|  |  |  |  |
| 8 | 1 | SearchLinear/Binary | 245,++ |
|  | 3 | SortingBubble/Selection |  |
|  | 5 | Applied toVectors |  |
|  |  |  |  |
| \*\*\*\*\*\*Not | requiredto | show |  |

**Program Code**:

/\*

File: main.cpp

Author: Bradley McKenzie

Created on February 6, 2017

Purpose: Project 2

\*/

//System Libraries

#include <iostream> //Standard library

#include <cstdlib> //Random Numbers/Exit

#include <ctime> //Time to set the seed

#include <fstream> //File stream library

#include <iomanip> //Format library

#include <string> //String library

#include <vector> //Vector library

using namespace std;

//User Libraries

//Global Constants

//Such as PI, Vc, -> Math/Science Values

//as well as conversions from system of units to another

const int PERCENT=100;//Calculate Wins and Losses Percentages

const int HUNDRDS=100;//Limit Number of Games 100's

const int TENS=10;//Limit to 10's

const int ROW=2; const int COL=1;//2D Array

//Function Prototypes

int percRes(float, float);

void respond(float);

bool wnta(float = 0);

void disHi();

void getUser();

bool findColor(string [], int , string);

//Executable code begins here!!!

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

//Set the random number seed

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

//Instantiate and Open files

ifstream in;

ofstream out;

ofstream hi;

in.open("NumberOfGames.dat");//Input Number of Games

out.open("ResultsOfGames.dat");//Output Result of Games

hi.open("HighScoreOfGames.dat",ios::app);//Output Score if Winnings > 0

//Declare Variables

float bet;//Bet Amount

float payout;//Pay Out Result

float sum = 0;//Total Winnings

float pwings = 0;//Possible Wins

float wins=0;//Number of Wins

float losses=0;//Number of Losses

const int SIZE=8;

const int COLOR=4;

string find;

string passwrd;//Password

string usrname;//Username

string bcolor;//Bet Color

string rcolor;//Result Color

string vcolor[SIZE]={"Red","red","RED","r","Blue","blue", "BLUE","b"};

string red[COLOR]={"Red","red","RED","r"};

string blue[COLOR]={"Blue","blue","BLUE","b"};

string van[ROW][COL]={{" Thank You, for Playing."},

{"A Game By Bradley McKenzie"}};//Initialize the 2-D Array

unsigned int nGames=3;//Standard Number of Games

//Input Values

cout<<" Double or Nothing"<<endl;

cout<<" A Game By Bradley McKenzie"<<endl<<endl;

cout<<"Enter Username: ";

getline(cin,usrname);//Get Username

do{

cout<<"Enter Password: ";

getline(cin,passwrd);//Get Correct Password

if(passwrd!="password")//If password is wrong ask for it again

cout<<" Incorrect Password, Please Try Again: ";

}while(passwrd!="password");cout<<endl;

while(in>>nGames);//Loop to end

nGames=nGames>HUNDRDS?TENS:nGames;//Limit the number of games

for(int game=1;game<=nGames;game++){//Loop the Game

cout<<"Enter the Amount of Your Bet: $";

cin>>bet;//Bet amount input

while(bet<=0){//Not allow bet to negative or 0

cout<<" Invalid: Bet Must be Greater Than $0."<<endl;

cout<<" Enter the Amount of Your Bet: $";

cin>>bet;//Bet amount input

}

//Input validation loop

cout<<"Place Bet On (Red or Blue): ";

cin>>bcolor;//Bet color input

bool valid = false;

for(int i=0;i<SIZE && !valid ;i++){//Check if bet color is valid

if (bcolor == vcolor[i]) valid = true;

}

//if !valid, then output invalid message

if(!valid){//If not valid ask for bet color again

cout<<" Invalid: Bet Color Must be (Red or Blue)"<<endl;

cout<<" Place Bet On (Red or Blue): ";

cin>>bcolor;//Bet color input

}

//end input validation loop

//Process by mapping inputs to outputs

//Last line will be number of games from file

//Call random number generator for the color landed on

int lcolor=rand()%26+1;//Value from 1 to 26 //Spin Wheel

if(lcolor % 2 == 0){//Land on Red

if(findColor( red, COLOR, bcolor))

{//Bet Won

rcolor="Red";

payout=bet\*2;//Double Bet

wins++;//Add plus one win

}

else if(findColor( blue, COLOR, bcolor))

{//Bet Loss

rcolor="Red";

payout=-bet;//Minus Bet

losses++;//Add plus one loss

}

else{//Invalid Input

rcolor="Invalid: Bet Color";

payout=0;

}

//Output Game Result for Land on Red

cout<<fixed<<setprecision(2);

cout<<" The spinner landed on: "<<rcolor<<endl;

cout<<" Your pay out is: $"<<payout<<endl;

out<<" The spinner landed on: "<<rcolor<<endl;

out<<" Your pay out is: $"<<payout<<endl;

}

else{//Land on Blue

if(findColor(blue, COLOR, bcolor)){//Bet Won

rcolor="Blue";

payout=bet\*2;//Double bet

wins++;//Add plus one win

}

else if(findColor(red, COLOR, bcolor)){//Bet Loss

rcolor="Blue";

payout=-bet;//Minus Bet

losses++;//Add plus one loss

}

else{//Invalid Input

rcolor="Invalid Bet Color";

payout=0;

}

//Output Game Result For Land on Blue

cout<<fixed<<setprecision(2);

cout<<" The spinner landed on: "<<rcolor<<endl;

cout<<" Your pay out is: $"<<payout<<endl;

out<<" The spinner landed on: "<<rcolor<<endl;

out<<" Your pay out is: $"<<payout<<endl;

}

sum += payout;//Get Sum of Pay Out

pwings += bet\*2;//Get Sum of Bet

cout<<" Current Winning: $"<<sum<<endl<<endl;//Display Current Winnings

}

if(sum>=0){//Output Username and Total Winnings if Winnings is > 0

hi<<"Username: "<<usrname<<" || Total Winnings: $"<<sum<<"\r\n";

}

hi.close();//Close High Score File

//Output Percentage of Wins and Losses

cout<<fixed<<setprecision(2);

out<<fixed<<setprecision(2);

cout<<" -----------------------------"<<endl;

cout<<" Percentage Wins of = "<<percRes(wins,nGames)<<"%"<<endl;//Percent Wins

cout<<" Percentage of Losses = "<<percRes(losses,nGames)<<"%"<<endl;//Percent Losses

cout<<" Possible Winnings: $"<<pwings<<endl;//Possible Winnings

cout<<" Total Winning: $"<<sum<<endl;//Actual Winnings

cout<<" -----------------------------"<<endl;

respond(sum);//Output Result Based on Winnings

wnta(sum);

out<<" -----------------------------"<<endl;

out<<" Percentage of Wins = "<<percRes(wins,nGames)<<"%"<<endl;//Percent Wins

out<<" Percentage of Losses = "<<percRes(losses,nGames)<<"%"<<endl;//Percent Losses

out<<" Possible Winnings: $"<<pwings<<endl;//Possible Winnings

out<<" Total Winning: $"<<sum<<endl;//Actual Winnings

out<<" -----------------------------"<<endl;

for(int row=0; row<ROW; row++){//Thank player for playing the game

for(int col=0;col<COL;col++){

cout<<" "<<van[row][col];//Output rows and column

}

cout<<endl;

}

cout<<endl;

cout<<" Show High Scores (Y or N): ";//Display High Score From File

disHi();//Get Y or N input then show high scores or exit

//Find a username in high score file

getUser();

//Close Files and Exit stage right!

in.close();

out.close();

return 0;

}

int percRes(float wl, float nGames){//Calculate Wins and Loss Percentage

return (int)(((float)PERCENT)\*wl/nGames);

}

void respond(float n){//Output a respond based on Winnings

if(n<=0){//Loss money respond

cout<<" Better Luck Next Time."<<endl;

}

else if(5000000>n && n>0){//Won money respond

cout<<"\t Nice Job!"<<endl;

}

else if(n>=5000000){//Won a lot of money respond

cout<<" Quitting Your Job?"<<endl;

}

}

bool wnta(float n){//Respond if player losses money

if(n<=0){

cout<<" Why Not Try Again?"<<endl;

}

return true;

}

void disHi(){//Display High Score Yes or No

string in;

char openHi;//Declare option to display high score

cin>>openHi;//Get Yes or No

if(openHi=='Y' || openHi=='y'){//Input Y or y to open high score

ifstream inFile("HighScoreOfGames.dat");

if(inFile){

while(!inFile.eof()){

//Input each line of the file

getline(inFile, in);

cout<<in<<endl;

}

}

}

else{

exit(0);

}

}

void getUser(){//Search for a username

int offset;

string line;

string find;

ifstream inFind("HighScoreOfGames.dat");//Input High Score File

cout<<"Enter Username to Search For: ";

getline(cin,find);

cin.ignore();

if(inFind.is\_open()){

while(!inFind.eof()){

getline(inFind,line);

if((offset = line.find(find,0)) != string::npos){

cout<<" The Word Has Been Found"<<find<<endl;

}

}

inFind.close();}//Close File

else

cout<<" Could Not Open File"<<endl;

}

bool findColor(string c[], int cSize, string test){ //Test Color

bool result = false;

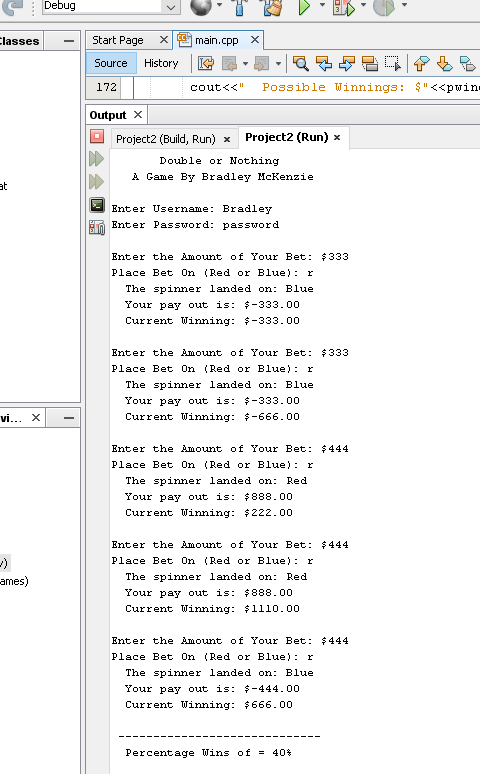
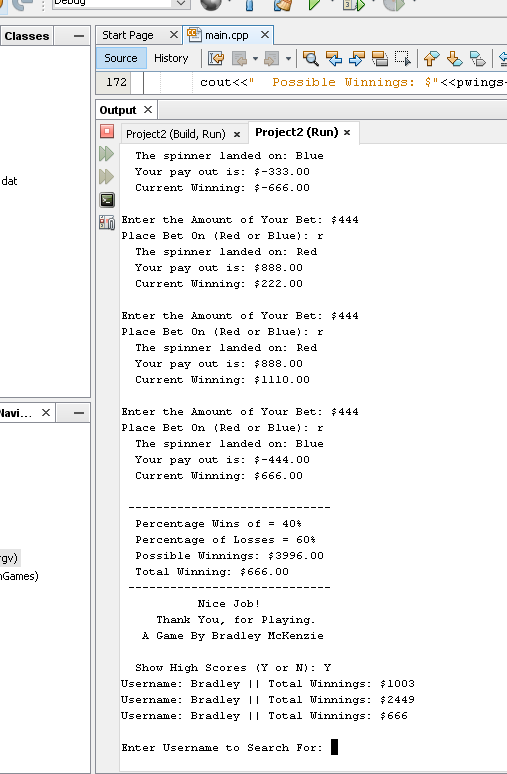
for (int i = 0; i < cSize ; i++){

if (c[i] == test) result = true;

}

return result;

}

**Sample Runs:**

