**Project 1**

# <Fifty>Introduction

Title: fifty

This is the dice game familiarly called Fifty.

The program utilizes two players in which they take turns rolling a pair of dice until one player has scored 50 points. The rules can be accessed at the beginning of the project if the user does not know how to play. The game is repeated until one player decides not to play again. The win and efficiency rates are displayed after each round in a display table.

## Summary

Project size: about 220 lines

The number of variables: about 30

This project includes almost all the concepts that we learned from the chapters we have studied so far in the textbook. I’m not sure if I can expand this project in the future but I believe I can change many of the calculations into functions and arrays which will make the code cleaner and legible.

This project took me about a week to complete. Each version of the project took me about a day and a half to complete. I attempted to incorporate more of the material we have learned in each rendition which complicated the coding experience and caused a multitude of problems when beginning an updated version.

I am satisfied with this project. I believe I have coded this project to the best of my abilities and am impressed with the outputs I have created, especially my “results display”.

I had a lot of fun writing this code and it was fun making a project of my own choice which allowed me to have lots of flexibility but made it difficult to manage at times.

## Game Play and Rules

This program will simulate the dice game Fifty which requires

2 players with each player taking turns to roll 2 die.

The goal of Fifty is to be the first player to reach 50 points.

You get points by rolling doubles.

All doubles except 3s and 6s score 5 points. Double 6s are worth

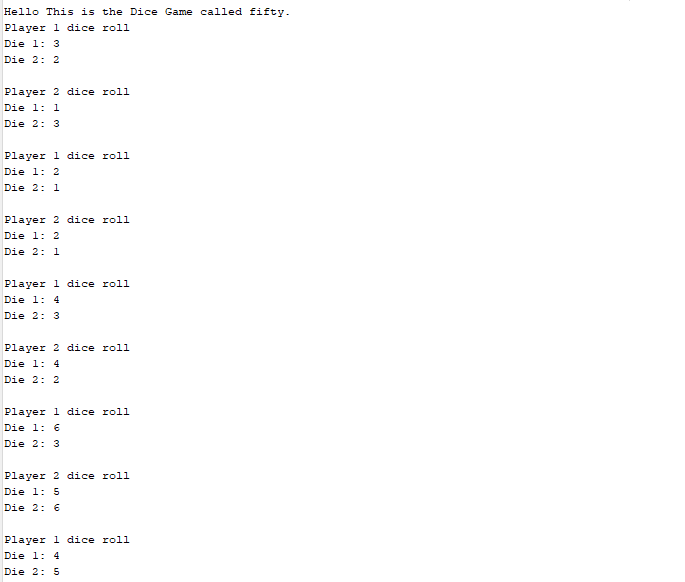
25 points and double 3s wipe out the players entire score

and the player must start again at 0. Non-double rolls are 0

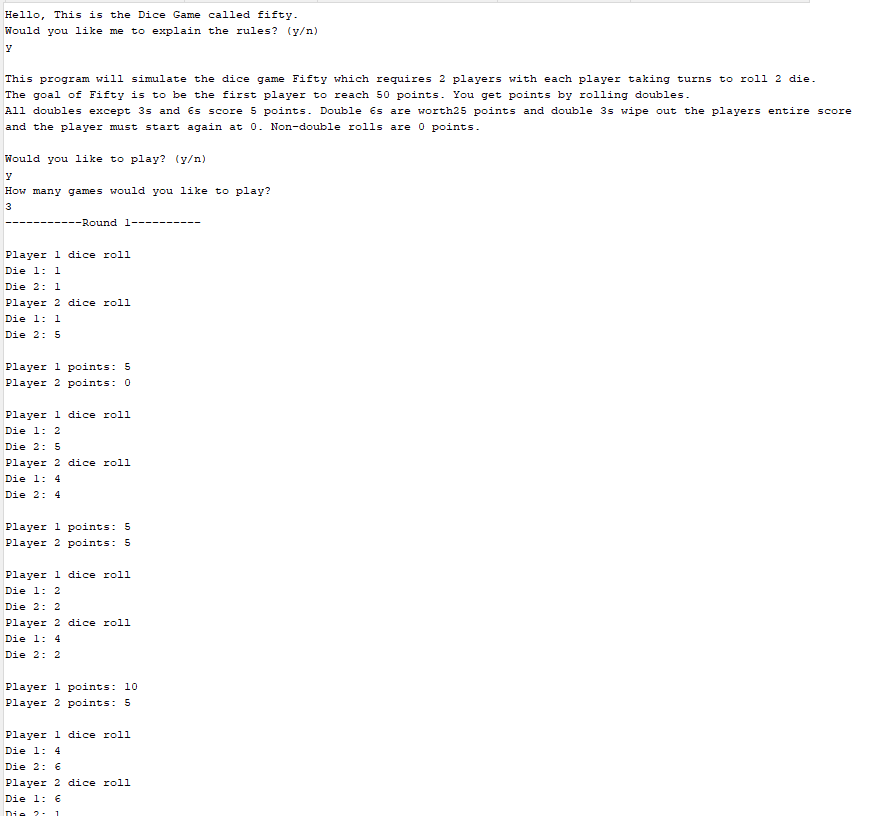
points.

## Development

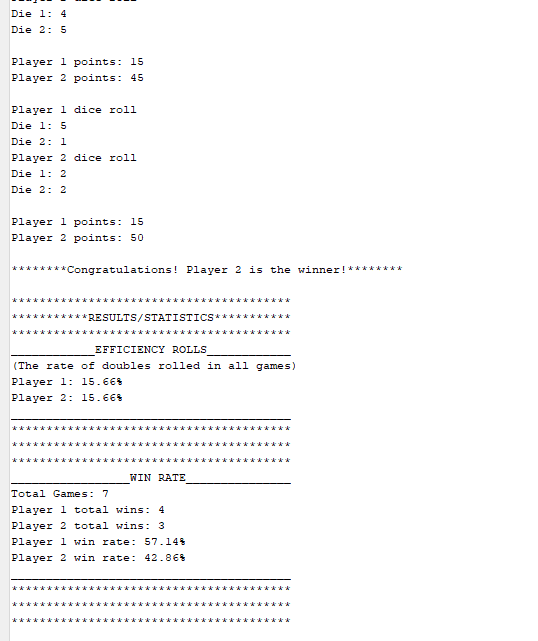
**Version 1.0**: My beginning version utilizes dice variables and score variables. This version does not utilize any user input but only focuses on the random initialization for the dice. This game is only run once and the score is revealed at the end of the program after all the dice have been rolled and one player has accomplished 50 points.



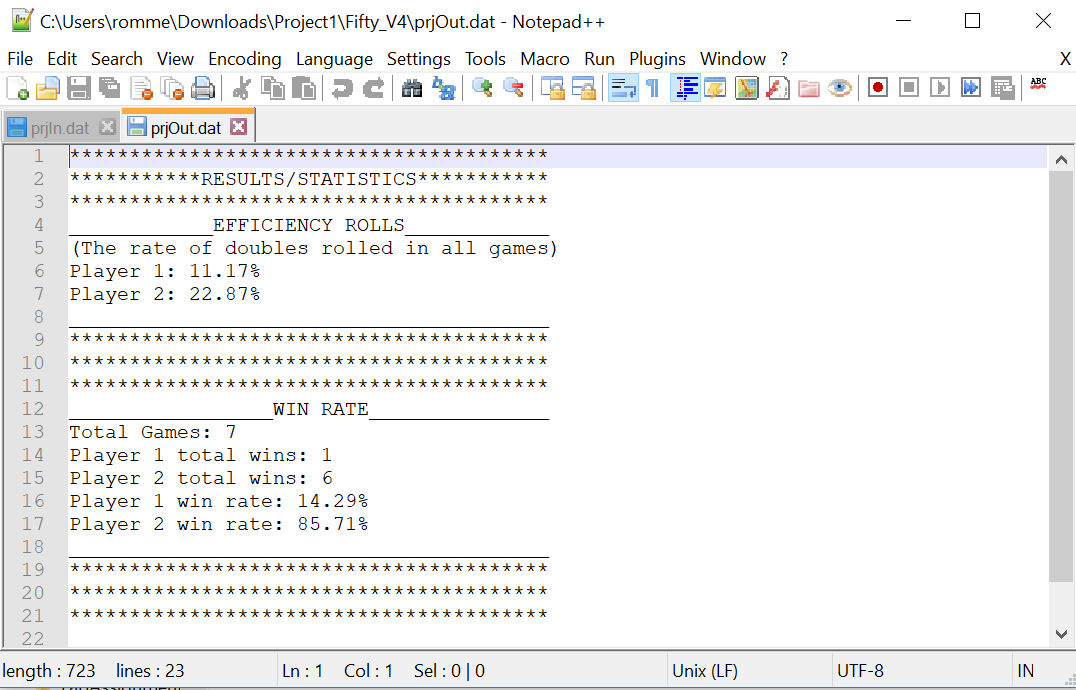
**Version 2.0:** This version is built on top of the initial version. The second version now incorporates instructions, play again feature and the amount of games. New variables were created for the adjustment. Now when the program runs, it asks the user whether or not they need instructions. After, the program now asks the user if they would like to play the game or not and asks for how many rounds and is continuously run until one of the users does not want to play.



**Version 3.0:** In this version, a win rate calculator is utilized to display the win rate, amount of wins, and the double roll efficiency of each player corresponding to how many rounds they initially set the game to. In addition when constructing whether or not the user needs the rules, it utilizes a bool operator rather than a comparison in the conditional.



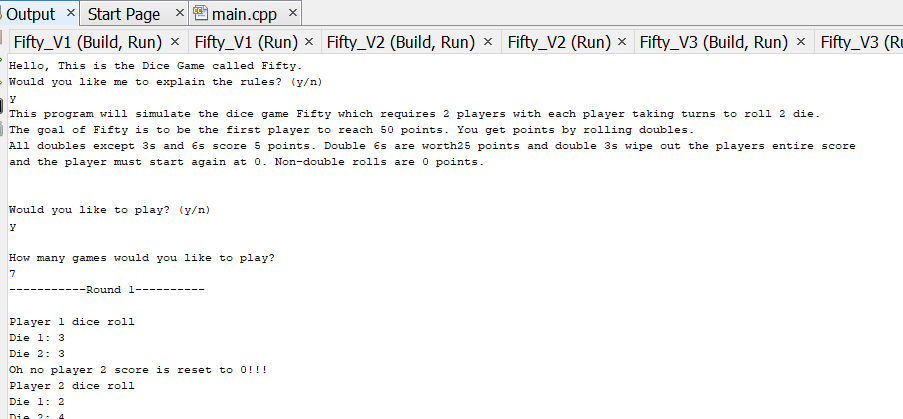
**Version 4.0:** Very similar to the version 3 of the program, however now incorporates input and output files. Instead of initializing variables on the main file, they are now initialized in the input file within the folders of the project. In addition to showing the display menu with the results and calculations of the win rate and efficiency rate, they are also now displayed onto a separate file in the project folder without the output of all the additional dice rolls and wins.



## Example inputs/outputs of final version

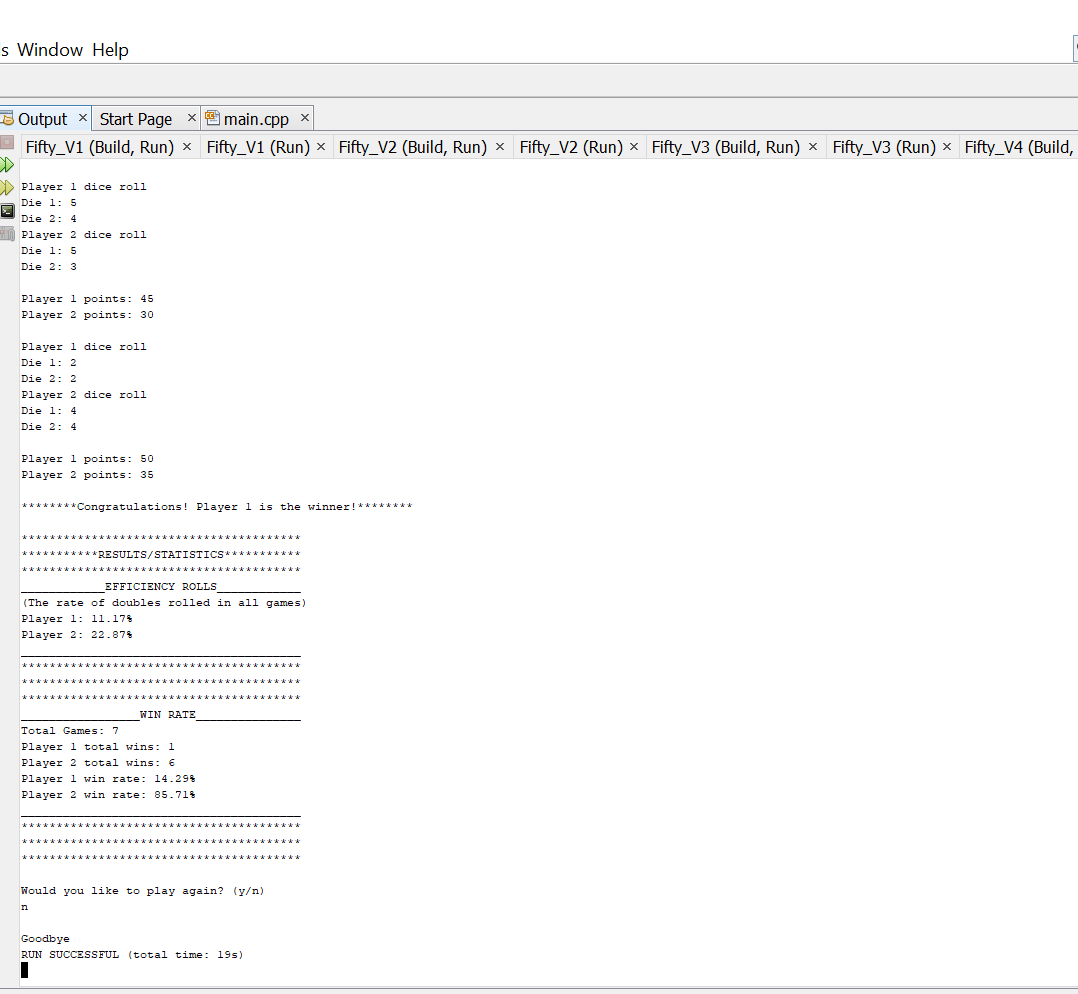
The game begins as soon as the program is run. It asks the user whether or not they need the rules, displays the rules when needed and continues when not.

The program now asks the user whether or not they want to play. If they want to play the game specifications begin, if not, will display a farewell output.



It will now ask the user how many games they would like to play. Then will display all the games and each roll the player and a scoreboard after every roll. In addition it will illustrate who won after each game and the results will be displayed at the end of the game.

It will then ask whether or not the set of players would like to play again, if not the program will stop and display a farewell image, if it does, it will repeat the same process until they do not want to play again.



## 

## Diagrams

### Pseudocode

*System Libraries*

*Declare/initialize*

*If needs rules*

*Output rules*

*While playing is yes*

*While score is less than 50*

*Roll Dice for player 1*

*Roll Dice for player 2*

*Add scores based on roll*

*Output Score*

*If player 1 has a greater score*

*Output Congratulations to player 1*

*Else*

*Output Congratulations to player 2*

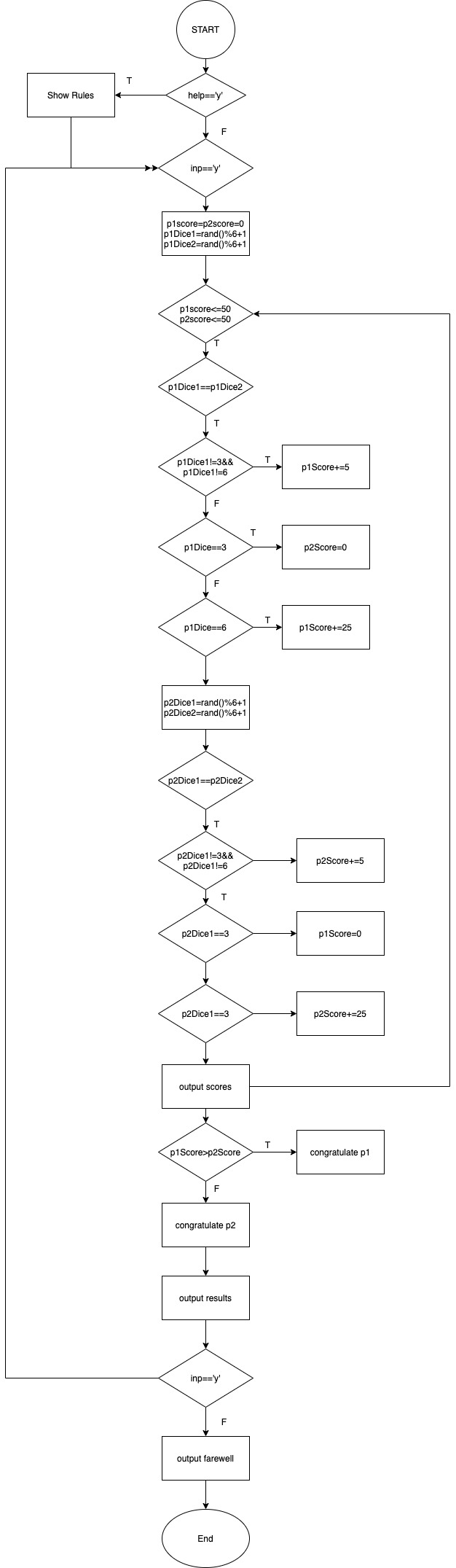
*Display Results*

*Ask to play again*

*Output Farewell*

### 

### Flowchart



## 

## Program Listing

**Fifty\_V4 (main.cpp)**

/\*

\* File: main.cpp

\* Author: Elijah De Vera

\* Date: February 02, 2021

\* Purpose: Project 1 Dice Game (fifty)

\* Version: 4

\*/

//Version 4 now implements input output files

//System Libraries - Post Here

#include <iostream> //input output library

#include <iomanip> //formatting

#include <cstdlib> //random srand to set the seed

#include <ctime> //time for rand

#include <cmath> //math functions

#include <fstream> //file I/O

**using** **namespace** std;

//Global Constants

**const** **float** PERCENT=100.0f; //conversion to percent;

//Execution Begins Here

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

cout<<"Hello, This is the Dice Game called Fifty."<<endl;

//Declare variables or constants here

//7 characters or less

ifstream in; //input file

ofstream out; //output file

**char** p1Dice1, // player 1 dice 1

p1Dice2, // player 1 dice 2

p2Dice1, // player 2 dice 1

p2Dice2, // player 2 dice 2

help, // player instructions

inp; // play again feature

**int** p1score, // player 1 score

p2score, // player 2 score

games, // amount of games

p1Doub, // player 1 double roll count

p2Doub; // player 2 double roll count

string rules,// string for rules

inFlNm, // input string file name

outFlNm;// output string file name

**float** p1Eff, // player 1 efficiency rate

p2Eff, // player 2 efficiency rate

turns, // amount of turns

p1w, // player 1 win count

p2w, // player 2 win count

p1wRate,// player 1 win rate

p2wRate;// player 2 win rate

**bool** read; //input for reading rules

//Uses current time as seed for random generator

srand(time(0)); //time the game play

//Initializing variables

rules = "This program will simulate the dice game Fifty which requires "

"2 players with each player taking turns to roll 2 die. \n"

"The goal of Fifty is to be the first player to reach 50 points. "

"You get points by rolling doubles. \n"

"All doubles except 3s and 6s score 5 points. Double 6s are worth"

"25 points and double 3s wipe out the players entire score \n"

"and the player must start again at 0. Non-double rolls are 0 "

"points.";

inFlNm="prjIn.dat"; //opening file name

outFlNm="prjOut.dat";

in.open(inFlNm); //file name opening

out.open(outFlNm); //file name opening

in>>p1Doub; //starting value for wins from input file

in>>p2Doub; //starting value for wins from input file

in>>turns; //turns amount from input file

//Asking user if they need instructions on how to play

cout<<"Would you like me to explain the rules? (y/n)"<<endl;

cin>>help; //input char for output rules

**if**(help=='y'){ //independent if

read=**true**; //setting char input into a boolean

}

**switch**(read){ //switch case for rules output

**case** 1: //reading if boolean is equal to true

cout<<rules<<endl<<endl;//outputs rules if true

**break**;

}

//Process inputs - map to outputs here

cout<<endl<<"Would you like to play? (y/n)"<<endl;

cin>>inp;

**while**( inp=='y' ){

cout<<endl<<"How many games would you like to play?"<<endl;

cin>>games; // input for amount of games

p1w=p2w=0; // resets win counter

**for** ( **int** i=1; i<=games; i++ ){ //will play amount of games user has input

cout<<"-----------Round "<<i<<"----------"<<endl<<endl;

//resets score each round

p1score=0;

p2score=0;

**do**{

cout<<"Player 1 dice roll"<<endl; // rolling first player pair of dice

p1Dice1=rand()%6+1;

p1Dice2=rand()%6+1;

cout<<"Die 1: "<<**static\_cast**<**int**>(p1Dice1)<<endl

<<"Die 2: "<<**static\_cast**<**int**>(p1Dice2)<<endl; //outputs die values

//dice rolls must equal to each other in order to score

**if** ( p1Dice1==p1Dice2 ){

p1Doub++;

**if** ( p1Dice1!=3 && p1Dice1!=6 ){ //doesnt roll a pair of 3 or 6

p1score+=5;

}

**else** **if** ( p1Dice1==3 ){ // rolls a pair of 3s

p2score=0;

cout<<"Oh no player 2 score is reset to 0!!!"<<endl;

}

**else** **if** ( p1Dice1==6 ){ // rolls a pair of 6s

p1score+=25;

}

}

cout<<"Player 2 dice roll"<<endl; // rolling first player pair of dice

p2Dice1=rand()%6+1;

p2Dice2=rand()%6+1;

cout<<"Die 1: "<<**static\_cast**<**int**>(p2Dice1)<<endl

<<"Die 2: "<<**static\_cast**<**int**>(p2Dice2)<<endl; //outputs die values

//dice rolls must equal to each other in order to score

**if** ( p2Dice1==p2Dice2 ){

p2Doub++;

**if** ( p2Dice1!=3 && p2Dice1!=6 ){ //doesnt roll a pair of 3 or 6

p2score+=5;

}

**else** **if** ( p2Dice1==3 ){ // rolls a pair of 3s

p1score=0;

cout<<"Oh no player 1 score is reset to 0!!!"<<endl;

}

**else** **if** ( p2Dice1==6 ){ // rolls a pair of 6s

p2score+=25;

}

}

//displays score every turn

cout<<endl;

cout<<"Player 1 points: "<<p1score<<endl;

cout<<"Player 2 points: "<<p2score<<endl<<endl;

turns++;

}**while** ((p1score<50 && p2score<50) && i<=games);

//displays the winner

**if** ( p1score>p2score ){

cout<<"\*\*\*\*\*\*\*\*"

"Congratulations! Player 1 is the winner!"

"\*\*\*\*\*\*\*\*"<<endl<<endl; //displays winner of that game

p1w++;

}

**else**{

cout<<"\*\*\*\*\*\*\*\*"

"Congratulations! Player 2 is the winner!"

"\*\*\*\*\*\*\*\*"<<endl<<endl; //displays winner of that game

p2w++;

}

}

//calculating efficiency of doubles for both pllayers

p1Eff=p1Doub/turns\*PERCENT;

p2Eff=p2Doub/turns\*PERCENT;

//calculating win rate for both players

p1wRate=p1w/games\*PERCENT;

p2wRate=p2w/games\*PERCENT;

//outputs for all statistics calculated

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

cout<<"\*\*\*\*\*\*\*\*\*\*\*RESULTS/STATISTICS\*\*\*\*\*\*\*\*\*\*\*"<<endl;

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

cout<<setprecision(2)<<fixed; //formatting for percentages

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_EFFICIENCY ROLLS\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

cout<<"(The rate of doubles rolled in all games)"<<endl;

cout<<"Player 1: "<<p1Eff<<"%"<<endl;

cout<<"Player 2: "<<p2Eff<<"%"<<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

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

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

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

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_WIN RATE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

cout<<"Total Games: "<<noshowpoint<<games<<endl;

cout<<"Player 1 total wins: "<<noshowpoint<<**static\_cast**<**int**>(p1w)<<endl;

cout<<"Player 2 total wins: "<<noshowpoint<<**static\_cast**<**int**>(p2w)<<endl;

cout<<"Player 1 win rate: "<<p1wRate<<"%"<<endl;

cout<<"Player 2 win rate: "<<p2wRate<<"%"<<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

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

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

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

//outputs for all statistics calculated to a file

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*RESULTS/STATISTICS\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<setprecision(2)<<fixed; //formatting for percentages

out<<"\_\_\_\_\_\_\_\_\_\_\_\_EFFICIENCY ROLLS\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

out<<"(The rate of doubles rolled in all games)"<<endl;

out<<"Player 1: "<<p1Eff<<"%"<<endl;

out<<"Player 2: "<<p2Eff<<"%"<<endl;

out<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_WIN RATE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

out<<"Total Games: "<<noshowpoint<<games<<endl;

out<<"Player 1 total wins: "<<noshowpoint<<**static\_cast**<**int**>(p1w)<<endl;

out<<"Player 2 total wins: "<<noshowpoint<<**static\_cast**<**int**>(p2w)<<endl;

out<<"Player 1 win rate: "<<p1wRate<<"%"<<endl;

out<<"Player 2 win rate: "<<p2wRate<<"%"<<endl;

out<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

out<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl;

//play again feature

cout<<"Would you like to play again? (y/n)"<<endl;

cin>>inp;

}

//message for the end of the game

cout<<endl<<"Goodbye";

//Exit stage left

in.close(); //closing files

out.close();//closing files

**return** 0;

}