**Project 2**

# <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 380 lines

The number of variables: about 30

The number of functions: 15

This project includes almost all the concepts that we learned from the chapters we have studied so far in the textbook. I’ve expanded from my first project with the implementation of arrays, vectors and functions which all made my program have more fluidity.

This project took me about 4 days 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 also saved the I/O files for the last version as it made things easier to read, so they are present again in the final version.

I am satisfied with this project. I believe I did really well in the time frame of the project. Completing this and the final was a difficult task but a fun challenge nonetheless.

I had a lot of fun writing this code and it was fun making the second version of the project but sometimes the functions could cause a lot of frustrations, but all was well when it all pieced together in the final rendition.

## 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

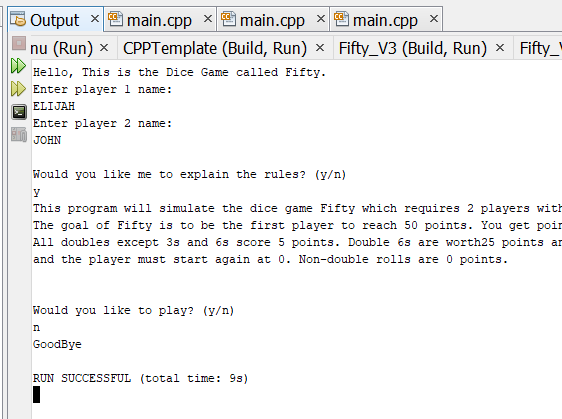
points.

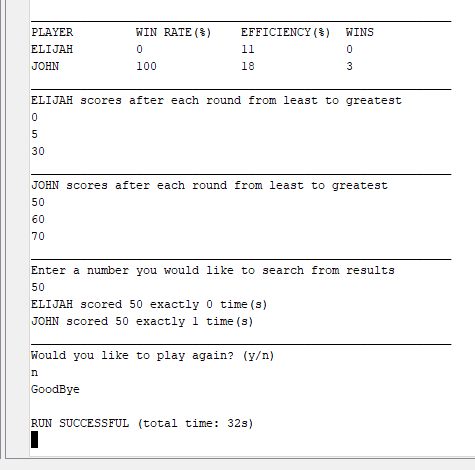
## Development

**Version 1.0**: My beginning version completely utilizes the same inputs and outputs as the final version of my final Project 1 version. However, the change is in the build. Instead of having all of my processing for outputs within the main function, I created new functions and implemented processes and calculations there.

### 

**Version 2.0:** This version further advances from the previous version. The program now prompts the players to enter their name and rather than displaying player 1 or player 2 in the outputs, it now displays the name they have entered into the program. In addition, this program now utilizes arrays, vectors, exit function, 2d arrays and overloading. Sorting and searching are also evident in this program which displays the scores from least to greatest and the user is able to enter a number they want to observe and the program searches for how many appearances that number has made in the final program.





**Version 3.0:** In the final version, I implemented a demo option which utilizes a default argument function and allows the user to see how playing the game is if they did not understand the rules completely. In addition, I re entered the I/O files and some of the inputs and outputs may be accessed or viewed on notepad++.

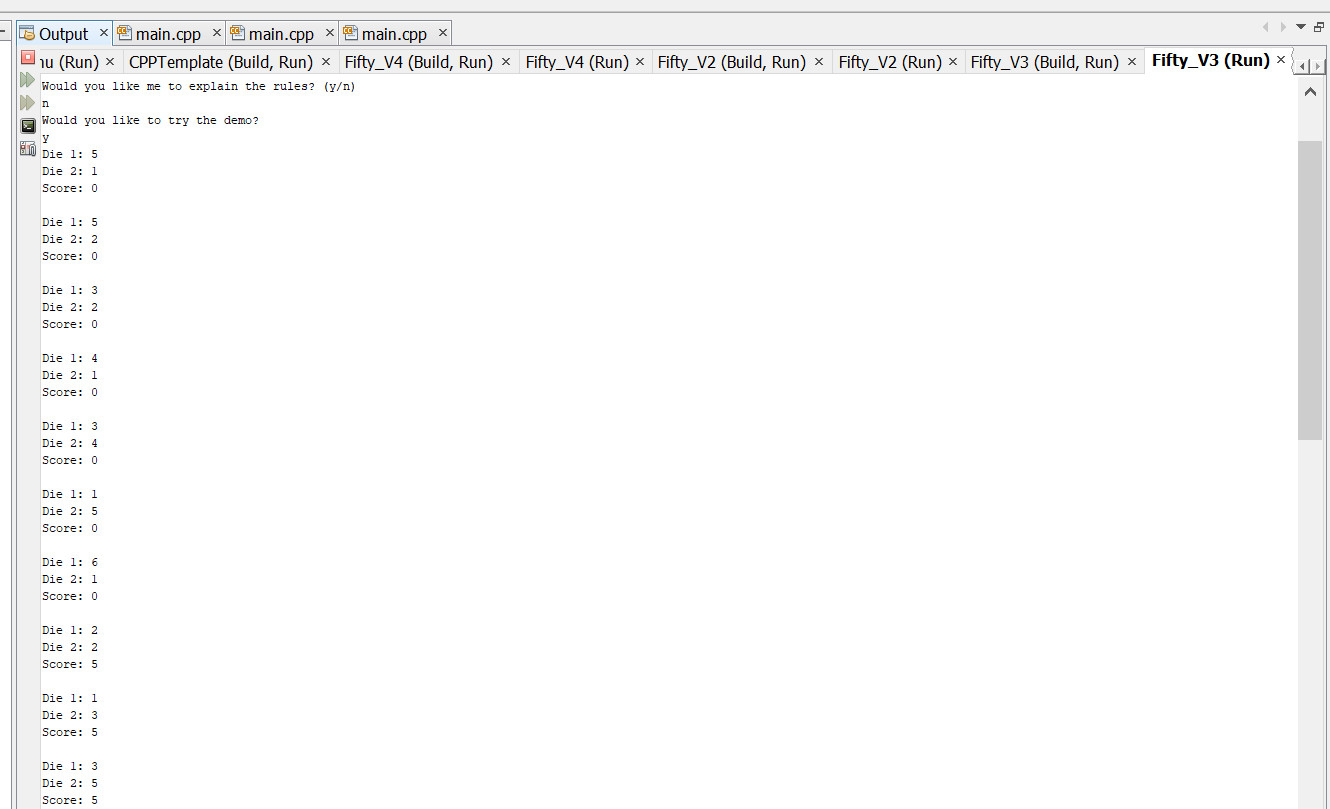
## 

## 

## Example inputs/outputs of final version

The game begins as soon as the program is run. It now asks for the players names and then asks the user whether or not they need the rules, or if they wish to see a demo.

After the demo is completed the program asks the user if they would like to play normally with a partner which inevitably keeps track of scores, rolls and other results.



## 

## 

## Diagrams

### Pseudocode

*System Libraries*

*Declare/initialize*

*If needs rules*

*Output rules*

*If wants demo*

*Output demo*

*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*

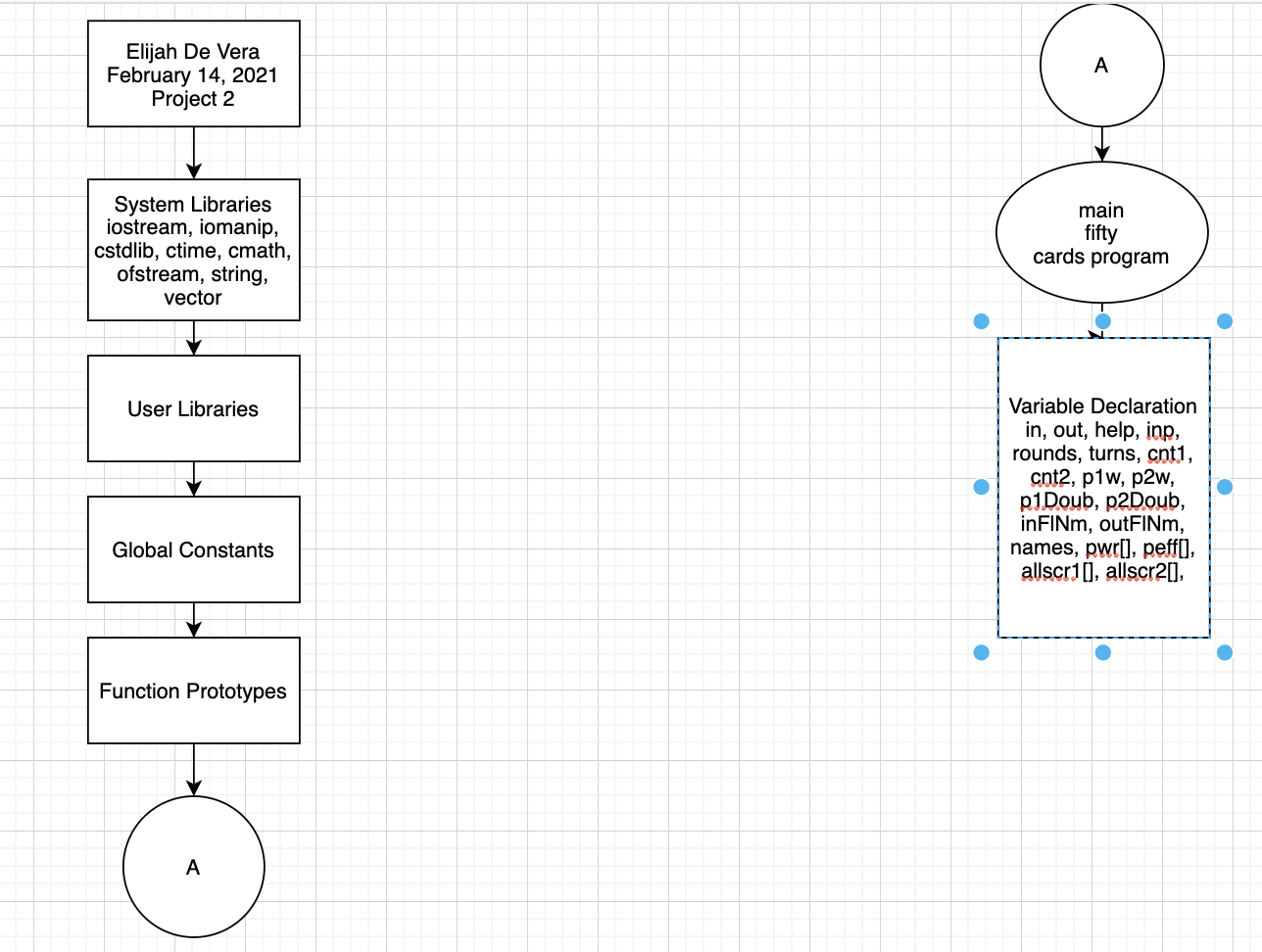
*Display Results*

*Ask to play again*

*Output Farewell*

### 

### Flowchart



## 

## 

## 

## My Approach

I wanted to first attack my original code by implementing outputs into functions and removing them from main. This was easily deemed the hardest part of the coding process for me. However, after completing this process creating and managing new concepts such as arrays and vectors were much easier.

After creating all the functions to manage inputs and outputs, I utilized arrays to handle player 1 and player 2 data. Instead of separating the two and creating two new variables, I created an array with only 2 parameters. The first parameter would always be for the first player and the second parameter would be for the second player.

Most of the arrays and functions are utilized to increase fluidity and to pass by data which makes the main function cleaner and easier to process.

## References

Dr. Lehr’s Lectures and lab

Class textbook

## Program Listing

**Fifty\_V5 (main.cpp)**

**/\***

**\* File: main.cpp**

**\* Author: Elijah De Vera**

**\* Date: February 13, 2021**

**\* Purpose: Project 2 Dice Game (fifty)**

**\* Version: 3**

**\*/**

**//Version 3 with defaults, 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**

**#include <string> //strings**

**#include <vector> //Vectors**

**using namespace std;**

**//Global Constants**

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

**//Function Prototypes**

**void pRules(char); //print rules**

**bool input(char); //handles inputs**

**void play(int,float&,float&,float&,float&,int&,vector<string>,**

**int[],int[]);//game process**

**int eff(int,float);//calculating player efficiency**

**int wr(float,int);//calculating player win rate**

**void data(vector<string>,float[],float[]);//calculating data**

**void getName(vector<string>&);//getting player names**

**void display(vector<string>,float[],float[], float, float);//displaying results**

**void print(string);//printing strings**

**void print(int);//printing ints**

**void print(float);//printing floats**

**void bubSort(int[],int,string);//bubble sort**

**void selSort(int[],int,string);//selection sort**

**void linSrch(int[],int[],vector<string>,int);**

**void demo(int=1);**

**//Execution Begins Here**

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

**//Heading**

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

**//Declare variables or constants here**

**//7 characters or less**

**const int PLAYERS=2;//only two players allowed**

**ifstream in; //input file**

**ofstream out; //output file**

**char help,**

**inp;**

**int rounds,**

**turns,**

**cnt1,**

**cnt2;**

**float p1w,**

**p2w,**

**p1Doub,**

**p2Doub;**

**string inFlNm,**

**outFlNm;**

**vector<string> names;**

**float pwr[PLAYERS];**

**float peff[PLAYERS];**

**int allScr1[10];**

**int allScr2[10];**

**//Initializing variables**

**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**

**//Uses current time as seed for random generator**

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

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

**getName(names);**

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

**cin>>help;**

**pRules(help);**

**//Asking if they want to try the demo**

**cout<<"Would you like to try the demo?"<<endl;**

**cin>>help;**

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

**demo();**

**cout<<endl;**

**}**

**//Process inputs - map to outputs here**

**cout<<"Would you like to play a real round with a partner? (y/n)"<<endl;**

**cin>>inp;//receiving input**

**while(input(inp)){**

**//Requesting for rounds**

**cout<<endl<<"How many rounds would you like to play?"**

**" (1-10)"<<endl;**

**cin>>rounds;**

**p1w=p2w=cnt1=cnt2=0;//resets every time user would like to play**

**play(rounds,p1w,p2w,p1Doub,p2Doub,turns,names,allScr1,allScr2);**

**peff[0]=eff(p1Doub,turns); //setting efficiency to player 1 data**

**peff[1]=eff(p2Doub,turns); //setting efficiency to player 2 data**

**pwr[0]=wr(p1w,rounds); //setting win rate to player 1 data**

**pwr[1]=wr(p2w,rounds); //setting win rate to player 2 data**

**//outputs for all statistics calculated**

**display(names,pwr,peff,p1w,p2w);**

**selSort(allScr1,rounds,names[0]);**

**bubSort(allScr2,rounds,names[1]);**

**linSrch(allScr1,allScr2,names,rounds);**

**//output file**

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

**out<<setw(15)<<left<<"PLAYER"<<setw(15)<<left<<"WIN RATE(%)"<<**

**setw(15)<<left<<"EFFICIENCY(%)"<<setw(15)<<left<<"WINS"<<endl;**

**out<<setw(15)<<left<<names[0]<<setw(15)<<left<<pwr[0]<<**

**setw(15)<<left<<peff[0]<<setw(15)<<left<<p1w<<endl;**

**out<<setw(15)<<left<<names[1]<<setw(15)<<left<<pwr[1]<<**

**setw(15)<<left<<peff[1]<<setw(15)<<left<<p2w<<endl;**

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

**//Play Again feature**

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

**cin>>inp;**

**input(inp);**

**}**

**//Closing Statement**

**cout<<endl<<"GoodBye"<<endl;**

**in.close(); //closing files**

**out.close();//closing file**

**return 0;**

**}**

**void linSrch(int a[], int b[], vector <string>names, int n){**

**int inp;**

**int cnt[2];**

**cnt[0]=cnt[1]=0;**

**cout<<"Enter a number you would like to search from results"<<endl;**

**cin>>inp;**

**for(int i=0;i<n;i++){**

**if(a[i]==inp){**

**cnt[0]++;**

**}if(b[i]==inp){**

**cnt[1]++;**

**}**

**}**

**for(int i=0;i<2;i++){**

**cout<<names[i]<<" scored "<<inp<<" exactly "<<cnt[i]<<" time(s)"<<endl;**

**}**

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

**}**

**void selSort(int a[], int n, string name){**

**cout<<name<<" scores after each round from least to greatest"<<endl;**

**for(int i=0;i<n-1;i++){**

**int idx=i;**

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

**if(a[idx]>a[j]){**

**idx=j;**

**}**

**}**

**int temp=a[i];**

**a[i]=a[idx];**

**a[idx]=temp;**

**}**

**for(int i=0;i<n;i++){**

**print(a[i]);**

**cout<<endl;**

**}**

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

**}**

**void bubSort(int a[], int n, string name){**

**cout<<name<<" scores after each round from least to greatest"<<endl;**

**int temp;**

**for(int i=0;i<n;i++){**

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

**if(a[j]<a[i]){**

**temp=a[i];**

**a[i]=a[j];**

**a[j]=temp;**

**}**

**}**

**}**

**for(int i=0;i<n;i++){**

**print(a[i]);**

**cout<<endl;**

**}**

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

**}**

**void print(float f){**

**cout<<f;**

**}**

**void print(int n){**

**cout<<n;**

**}**

**void print(string str){**

**cout<<str;**

**}**

**void display(vector<string> names, float pwr[],**

**float peff[], float p1w, float p2w){**

**//Heading**

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

**cout<<setw(15)<<left<<"PLAYER"<<setw(15)<<left<<"WIN RATE(%)"<<**

**setw(15)<<left<<"EFFICIENCY(%)"<<setw(15)<<left<<"WINS"<<endl;**

**cout<<setw(15)<<left<<names[0]<<setw(15)<<left<<pwr[0]<<**

**setw(15)<<left<<peff[0]<<setw(15)<<left<<p1w<<endl;**

**cout<<setw(15)<<left<<names[1]<<setw(15)<<left<<pwr[1]<<**

**setw(15)<<left<<peff[1]<<setw(15)<<left<<p2w<<endl;**

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

**}**

**void getName(vector<string>& names){**

**string name;**

**for(int i=0;i<2;i++){**

**cout<<"Enter player "<<i+1<<" name: "<<endl;**

**cin>>name;**

**names.push\_back(name);**

**}**

**cout<<endl;**

**}**

**int wr(float pw, int rounds){**

**int pwr;**

**pwr=pw/rounds\*PERCENT;**

**return pwr;**

**}**

**int eff(int wins, float turns){**

**int peff;**

**peff=wins/turns\*PERCENT;**

**return peff;**

**}**

**void demo(int rounds){**

**//Declaring variables**

**int score;**

**char dice1,**

**dice2;**

**//Initializing variables**

**score=0;**

**//Processing outputs**

**while(score<50){**

**for(int i=0;i<rounds;i++){**

**dice1=rand()%6+1;**

**dice2=rand()%6+1;**

**cout<<"Die 1: "<<static\_cast<int>(dice1)<<endl**

**<<"Die 2: "<<static\_cast<int>(dice2)<<endl; //outputs die values**

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

**if ( dice1==dice2 ){**

**if ( dice1!=3 && dice2!=6 ){ //doesn't roll a pair of 3 or 6**

**score+=5;**

**}**

**else if ( dice1==6 ){ // rolls a pair of 6s**

**score+=25;**

**}**

**}**

**cout<<"Score: "<<score<<endl<<endl;**

**}**

**}**

**cout<<"YOU WIN THE DEMO"<<endl;**

**}**

**void play(int rounds, float& p1w, float& p2w,**

**float& p1Doub, float& p2Doub, int& turns,**

**vector<string> names, int allScr1[],**

**int allScr2[]){**

**//Declaring Variables**

**const int PLAYERS=2;**

**const int ROLLS=2;**

**int scores[PLAYERS];**

**char dice[PLAYERS][ROLLS]; //2D array for dice rolls**

**//Processing Outputs**

**if(rounds>0){**

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

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

**//resets score each round**

**scores[0]=0;**

**scores[1]=0;**

**do{**

**cout<<names[0]<<"'s roll"<<endl; // rolling first player pair of dice**

**dice[0][0]=rand()%6+1;**

**dice[0][1]=rand()%6+1;**

**cout<<"Die 1: "<<static\_cast<int>(dice[0][0])<<endl**

**<<"Die 2: "<<static\_cast<int>(dice[0][1])<<endl; //outputs die values**

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

**if ( dice[0][0]==dice[0][1] ){**

**p1Doub++;**

**if ( dice[0][0]!=3 && dice[0][0]!=6 ){ //doesn't roll a pair of 3 or 6**

**scores[0]+=5;**

**}**

**else if ( dice[0][0]==3 ){ // rolls a pair of 3s**

**scores[1]=0;**

**cout<<"Oh no "<<names[1]<<" score is reset to 0!!!"<<endl;**

**}**

**else if ( dice[0][0]==6 ){ // rolls a pair of 6s**

**scores[0]+=25;**

**}**

**}**

**cout<<names[1]<<"'s roll"<<endl; // rolling first player pair of dice**

**dice[1][0]=rand()%6+1;**

**dice[1][1]=rand()%6+1;**

**cout<<"Die 1: "<<static\_cast<int>(dice[1][0])<<endl**

**<<"Die 2: "<<static\_cast<int>(dice[1][0])<<endl; //outputs die values**

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

**if ( dice[1][0]==dice[1][1] ){**

**p2Doub++;**

**if ( dice[1][0]!=3 && dice[1][1]!=6 ){ //doesnt roll a pair of 3 or 6**

**scores[1]+=5;**

**}**

**else if ( dice[1][0]==3 ){ // rolls a pair of 3s**

**scores[0]=0;**

**cout<<"Oh no "<<names[0]<<" score is reset to 0!!!"<<endl;**

**}**

**else if ( dice[1][0]==6 ){ // rolls a pair of 6s**

**scores[1]+=25;**

**}**

**}**

**//displays score every turn**

**//parallel array**

**cout<<endl;**

**for(int i=0;i<PLAYERS;i++){**

**cout<<names[i]<<"'s points: "<<scores[i]<<endl;**

**}**

**turns++;**

**}while ((scores[0]<50 && scores[1]<50) && i<=rounds);**

**//displays the winner**

**allScr1[i-1]=scores[0];**

**allScr2[i-1]=scores[1];**

**if ( scores[0]>scores[1] ){**

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

**"Congratulations! "<<names[0]<<" is the winner!"**

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

**p1w++;**

**}**

**else{**

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

**"Congratulations! "<<names[1]<<" is the winner!"**

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

**p2w++;**

**}**

**}**

**}else{**

**cout<<"ERROR! MUST ENTER A NUMBER GREATER THAN 0!";**

**exit(0);**

**}**

**}**

**bool input(char inp){**

**bool yes;**

**(inp=='y')?yes=true:yes=false;**

**return yes;**

**}**

**void pRules(char inp){**

**bool read;**

**string 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.";**

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

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

**}else{**

**read=false;**

**}**

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

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

**print(rules);**

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

**break;**

**}**

**}**