**Project 2**

Title

**War!**

**Card Game**

Course

**CSC-5**

Section

**40771**

Due Date

**February 14, 2020**

Author

**Christopher Mathis**

**Introduction**

Title: War!

This is the well-known card game War! very popular among children, known for its’ easy to follow and straight forward ruleset. It consists of usually two players whom each have 26 cards (half a deck of cards), the players are to not look at the cards; they are to proceed by both flipping the top card over. Numbered cards have their value and the face card values are jack(11), queen(12) and king(13); aces can either be high(14) or low(1), in this version aces are low. Whoever had the higher ranked card, wins that round and gets the losers’ card (suit does not matter in this game, neither does color. However if both players flip the same rank, then the game proceeds to War!, where each player puts down three cards face down and then flips another card from their deck. Whoever had the higher rank, wins the war; if the players flip the same rank again then War! proceeds again in a loop until a winner is declared. In the extremely unlikely event that a war proceeds until both players only have one card left to flip each then they would flip a coin to see who wins the game. The game ends when one player has all fifty-two cards and conversely the other player has zero.

**Summary**

Project Size: About 500 lines

The Number of Variables: 24

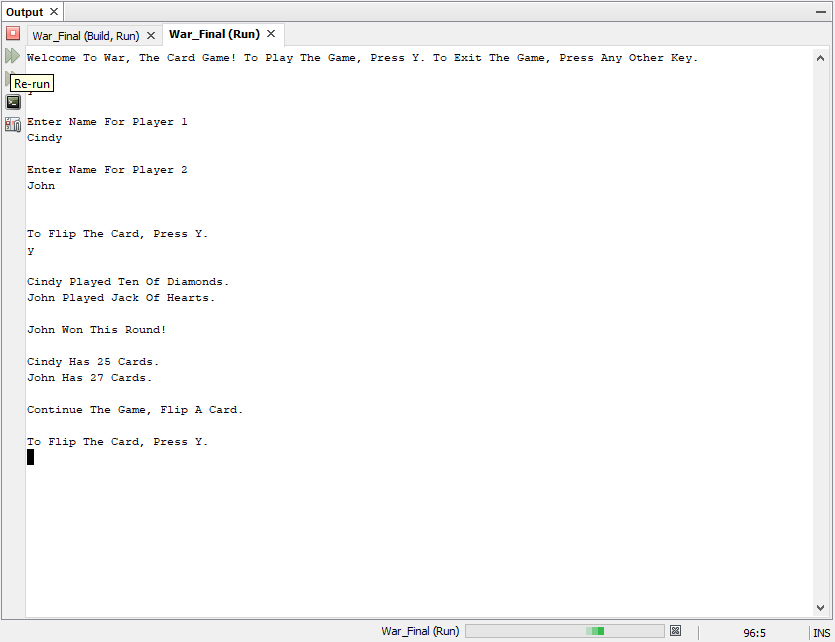
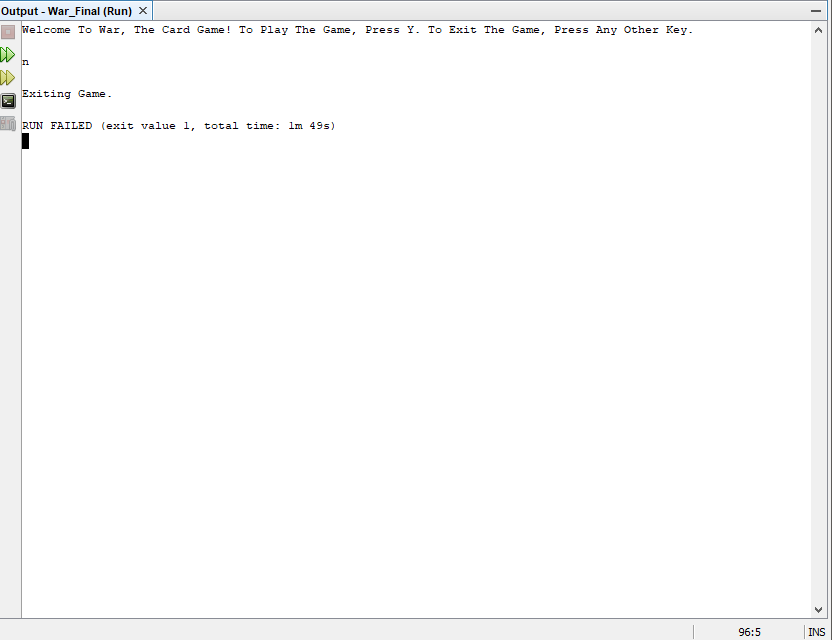
The Number of Concepts: 46

In this version of War! the game is fully functional, the only thing excluded is the ability to not have repeated cards, due to using a random number generator; I attempted to use files like the example provided in lab of unique card pull, but unfortunately that doesn’t allow for the correct comparison of the card ranks (some suits would be higher no matter the rank). I also attempted to use arrays/vectors but the I ran into the same issue.

I implemented as many concepts as I could, only things I could not figure out how to implement was the math library (the only computation was for percentage) and sorting/searching. This game just does not need them to function, if I would have implemented them, they would have been redundant and pretty much useless.

It took about 3 days (around 20 hours) this includes the thought process/brainstorming as well the actual coding, flowcharting, and write-up. I still ran into the same repeating card issue on this final version, but the game functions well enough as is. In later classes, I expect to learn concepts that would allow for m to fix that issue.

I did run into something interesting, if you enter an input too quickly, the card values would stay the same and not change, even though I have a random number generator constantly being looped.



**Cross Reference from Project 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chapter** | **Section** | **Topic** | **Where Line #''s** | **Pts** | **Notes** |
| 2 | 2 | cout | 67 |  |  |
|  | 3 | libraries | 9-15 | 5 | iostream, iomanip, cmath, cstdlib, fstream, string, ctime |
|  | 4 | variables/literals | 49-58 |  | No variables in global area, failed project! |
|  | 5 | Identifiers | 51 |  |  |
|  | 6 | Integers | 53 | 1 |  |
|  | 7 | Characters | 52 | 1 |  |
|  | 8 | Strings | 55 | 1 |  |
|  | 9 | Floats No Doubles | 54 | 1 | Using doubles will fail the project, floats OK! |
|  | 10 | Bools | 118 | 1 |  |
|  | 11 | Sizeof \*\*\*\*\* |  |  |  |
|  | 12 | Variables 7 characters or less | 49-58 |  | All variables <= 7 characters |
|  | 13 | Scope \*\*\*\*\* No Global Variables |  |  |  |
|  | 14 | Arithmetic operators | 97 |  |  |
|  | 15 | Comments 20%+ | 70 | 2 | Model as pseudo code |
|  | 16 | Named Constants | 22 |  | All Local, only Conversions/Physics/Math in Global area |
|  | 17 | Programming Style \*\*\*\*\* Emulate |  |  | Emulate style in book/in class repositiory |
|  |  |  |  |  |  |
| 3 | 1 | cin | 120 |  |  |
|  | 2 | Math Expression | 227 |  |  |
|  | 3 | Mixing data types \*\*\*\* |  |  |  |
|  | 4 | Overflow/Underflow \*\*\*\* |  |  |  |
|  | 5 | Type Casting | 98 | 1 |  |
|  | 6 | Multiple assignment \*\*\*\*\* |  |  |  |
|  | 7 | Formatting output | 102 | 1 |  |
|  | 8 | Strings | 75 | 1 |  |
|  | 9 | Math Library | N/A | 1 | All libraries included have to be used |
|  | 10 | Hand tracing \*\*\*\*\*\* |  |  |  |
|  |  |  |  |  |  |
| 4 | 1 | Relational Operators | 89 |  |  |
|  | 2 | if | 72 | 1 | Independent if |
|  | 4 | If-else | 122-125 | 1 |  |
|  | 5 | Nesting | 72-74 | 1 |  |
|  | 6 | If-else-if | 89-92 | 1 |  |
|  | 7 | Flags \*\*\*\*\* |  |  |  |
|  | 8 | Logical operators | 122 | 1 |  |
|  | 11 | Validating user input | 166 | 1 |  |
|  | 13 | Conditional Operator | 141 | 1 |  |
|  | 14 | Switch | 176 | 1 |  |
|  |  |  |  |  |  |
| 5 | 1 | Increment/Decrement | 225 | 1 |  |
|  | 2 | While | 66 | 1 |  |
|  | 5 | Do-while | 135-143 | 1 |  |
|  | 6 | For loop | 74 | 1 |  |
|  | 11 | Files input/output both | 64-65 | 2 |  |
|  | 12 | No breaks in loops \*\*\*\*\*\* |  |  | Failed Project if included |
| \*\*\*\*\*\* Not | required to | show | Total | 30 |  |

**Cross Reference for Project 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chapter** | **Section** | **Topic** | **Where Line #''s** | **Pts** | **Notes** |
| 6 |  | Functions |  |  |  |
|  | 3 | Function Prototypes | 25-44 | 4 | Always use prototypes |
|  | 5 | Pass by Value | 137 | 4 |  |
|  | 8 | return | 293 | 4 | A value from a function |
|  | 9 | returning boolean | 126 | 4 |  |
|  | 10 | Global Variables |  | XXX | Do not use global variables -100 pts |
|  | 11 | static variables | N/A | 4 |  |
|  | 12 | defaulted arguments | 426 | 4 |  |
|  | 13 | pass by reference | 146 | 4 |  |
|  | 14 | overloading | 26 & 30 | 5 |  |
|  | 15 | exit() function | 124 | 4 |  |
|  |  |  |  |  |  |
| 7 |  | Arrays |  |  |  |
|  | 1 to 6 | Single Dimensioned Arrays | 57 | 3 |  |
|  | 7 | Parallel Arrays | 57 & 58 | 2 |  |
|  | 8 | Single Dimensioned as Function Arguments | 26 | 2 |  |
|  | 9 | 2 Dimensioned Arrays | N/A | 2 | Emulate style in book/in class repositiory |
|  | 12 | STL Vectors | 58 | 2 |  |
|  |  | Passing Arrays to and from Functions | 223-238 | 5 |  |
|  |  | Passing Vectors to and from Functions | 241-255 | 5 |  |
|  |  |  |  |  |  |
| 8 |  | Searching and Sorting Arrays |  |  |  |
|  | 3 | Bubble Sort | N/A | 4 |  |
|  | 3 | Selection Sort | N/A | 4 |  |
|  | 1 | Linear or Binary Search | N/A | 4 |  |
| \*\*\*\*\*\* Not | required to | show | Total | 70 | Other 30 points from Project 1 first sheet tab |

**Major Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Variable Name** | **Description** | **Location** |
| int | pRank | Player 1 card rank | Main() |
|  | pSuit | Player 1 card suit | Main() |
|  | cRank | Player 2 card rank | Main() |
|  | cSuit | Player 1 card suit | Main() |
|  | SIZE | Size of array | Main() |
|  | Counter | Array | Main() |
|  | A | Counter | whoWon() |
|  | B | counter | whoWon() |
|  | C | Counter | whoWon() |
|  | D | Counter | whoWon() |
| Unsigned int | coin | Coin value | flipCoin() |
| char | Choice | User input | Main() |
|  | flip | User input for coin value | flipCoin() |
| fstream | In | In from file | Main() |
|  | Out | Out to file | Main() |
| float | pPerc | Percentage of player 1 wins | Main() |
|  | cPerc | Percentage of player 2 wins | Main() |
| string | Cnt | Statement to continue or game over (ternary operator) | Main() |
|  | S | Input for vector | Main() |
|  | W | Input for vector | Main() |
|  | Msg | Opening message to user | Main() |
|  | Name | vector | Main() |

**Pseudo Code**

*Comments (Name, Date, Etc)*

*System Libraries*

*Function Prototypes*

*Percentage Conversion*

*Main() Execution Begins*

*Declare Variables*

*Initialize Variables*

*Display Opening Message Using Files*

*While Loop to Output Whole File*

*Boolean Test To Start or Exit Game*

*If choice isn’t Y or y*

*Statements*

*Exit()*

*Else*

*Return true*

*If(true)*

*For loop to input user names into vector*

*Start the game loop*

*Initialize*

*Do-While Loop To Continue Game*

*Prompt To Flip Cards*

*While Loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Cards Are Greater Than 4*

*Statements*

*Prompt To Flip Cards*

*While Loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Player Has 4 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Computer Has 4 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Player Has 3 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Computer Has 3 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Player Has 2 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Computer Has 2 Cards*

*Statements*

*Prompt To Flip*

*While loop To Validate Input*

*Flip Cards*

*If Player Rank Is Greater*

*Statements*

*Counters*

*Else if Computer Rank Is Greater*

*Statements*

*Counters*

*Else*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Player Has 1 Card*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counters*

*Else if Player Rank Equals Computer Rank & Computer Has 1 Card*

*Statements*

*Prompt To Flip Coin*

*While Loop To Validate Input*

*Flip Coin*

*If Win Coin Flip*

*Statements*

*Counters*

*Else*

*Statements*

*Counter*

*Ternary Operator For Continuing Game Or Game Over Statements*

*While Loop To Continue Game (Less Than 52 Cards)*

*If Player Card Equals 52*

*Statement*

*If Computer Card Equals 52*

*Statement*

*Calculate Percentages of Wins*

*Statements Of Stats*

*Output Stats To File*

*Close Files*

*Return (Exit Program)*

**Flowhart**

Comments

Name

Date

Purpose

Libraries

iostream

cstdlib

ctime

fstream

iomanip

string

Global Constant

CNVPERC=100

Function Prototype

isTrue

whoWon

flipCard

cards

whoWon(2)

lesser

equal

equal1

equal2

equal3

equal4

equal5

equal5

equal7

equal8

equal9

equal10

equal11

main()

Declare

pRank=pSuit=

cRank=cSuit=0

counter={26,26,0,0}

Declare

in and out file streams

and open

A

in>>msg

True

Opening Message From

File

counter[0]==52

counter[1]==52

Output

name[2]

Output

name[3]

True

True

pPerc=pWIn/nRounds\*100

cPerc=cWin/nRounds\*100

Output

Game Stats

Output Game Stats To File

pWin, cWin, Perk, clerk

Close Files

Return

isTrue

true

int i=0

i<2

i++

True

Output

Enter Names

Input

getline

Close Files

return 0

True

whoWon

A

flipCard

set random number

seed

input

choice

validate

pRank=rand()

pSuit=rand()

cRank=rand()

cSuit=rand()

cards

validate

choice!=Y

&&

choice!=y

Output

invalid entry

input

choice

True

return

return

isTrue

whoWon

Declare

a=b=c=d=0

cnt

counter[0]>0

&

counter[1]>0

flipCard

whoWon(2)

lesser

equal

counter[0]>0

Output

Continue Game

Game Over

return

True

Input

choice

choice!=Y

&7

choice!=y

Output

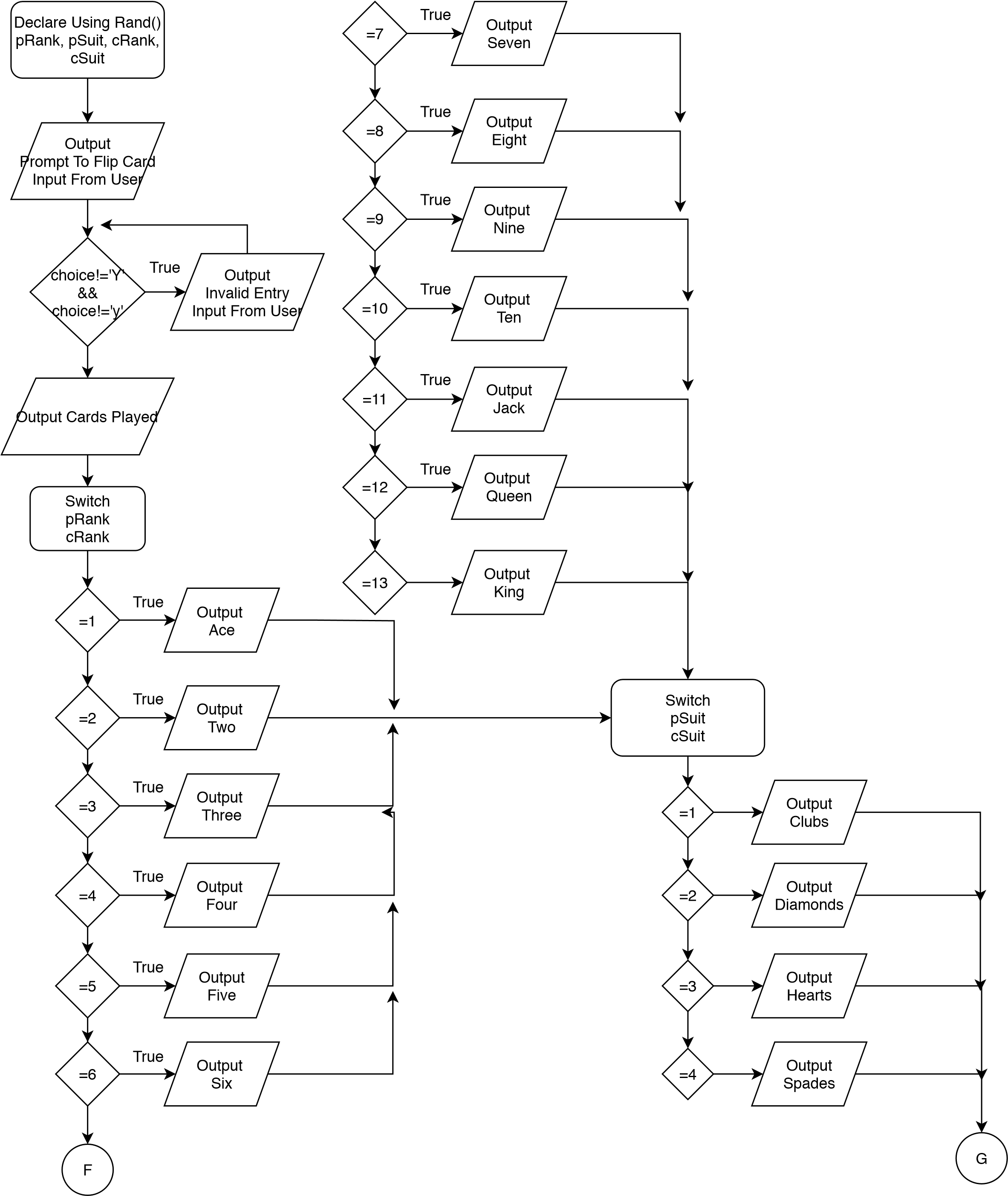
Exiting Game

True

exit()

return true

cards



**Program**

/\*

\* File: main.cpp

\* Author: Christopher Mathis

\* Created on February 14, 2020

\* Purpose: War Final Version

\*/

//System Libraries

#include <iostream> //Input/Output

#include <iomanip> //Format

#include <ctime> //Time

#include <cstdlib> //Srand/Rand

#include <string> //Strings

#include <fstream> //File I/O

#include <vector> //Vectors

using namespace std; //Standard Name-Space Where System Libraries Exist

//User Libraries

//Global Constants - No Global Variables

//Only Universal Constants, Math, Physics, Conversions, Higher Dimensions

const float CNVPERC=100;//Conversion From Decimal To Percent

//Function Prototypes

bool isTrue(char);

int whoWon(int, int, int, int, vector<string>, int []);

void flipCard(int &, int &, int &, int &, vector<string>);

void validate(char);

void cards(int, int, int, int,vector<string>);

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

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

int equal(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal1(int &, int &, int &, int &, vector<string>,int [], int, int);

int equal2(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal3(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal4(int &, int& , int &, int &, vector<string>, int [], int, int);

int equal5(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal6(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal7(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal8(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal9(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal10(int &, int &, int &, int &, vector<string>, int [], int, int);

int equal11(int &, int &, int &, int &, vector<string>, int [], int, int);

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

//Execution Begins Here

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

//Declare Variable Data Types and Constants

ifstream in; //In From File

ofstream out; //Out To File

string msg; //String To Use For Input From File

char choice; //Input From User To Play/Continue

int pRank, pSuit, cRank, cSuit; //Rank and Suit Of Cards

float pPerc, cPerc; //Percentage Of Rounds Won

string cnt; //Statement Variable Used With Ternary Operator

const int SIZE=4; //Size Of Array

int counter[SIZE]={26, 26, 0, 0}; //Array For Counters

vector<string> name; //Vector For Player Names

//Initialize Variables

pRank=pSuit=cRank=cSuit=0;

//Display Beginning Message Using File

in.open("start.dat");

out.open("stats.dat");

while(in>>msg){ //While Loop To Output Everything From File

cout<<msg<<" ";

}

//Boolean Test To Start Or Exit Game

isTrue(choice);

if(true){

cin.ignore();

for(int i=0; i<2; i++){ //For Loop To Input User Names Into Vector

string s;

cout<<"Enter Name For Player "<<i+1<<endl;

getline(cin,s);

name.push\_back(s);

cout<<endl;

}

string w=" Won";

name.push\_back(name[0]+w);

name.push\_back(name[1]+w);

//Start The Game Loop

whoWon(pRank, pSuit, cRank, cSuit, name, counter);

//Display The Winner

if(counter[0]==52){

cout<<endl<<name[2]<<" :)\n"<<endl;

}

else if(counter[1]==52){

cout<<endl<<name[3]<<" :)\n"<<endl;

}

}

pPerc=(static\_cast<float>(counter[2])/(counter[2]+counter[3]))\*CNVPERC;

cPerc=(static\_cast<float>(counter[3])/(counter[2]+counter[3]))\*CNVPERC;

//Output Stats To Player In Program

cout<<name[2]<<" "<<counter[2]<<" Rounds, Which Is Good For "

<<fixed<<setprecision(2)<<showpoint<<pPerc<<"% Of The Rounds Played.\n";

cout<<name[3]<<" "<<counter[3]<<" Rounds, Which Is Good For "

<<fixed<<setprecision(2)<<showpoint<<cPerc<<"% Of The Rounds Played.\n";

//Output Stats To File

out<<name[2]<<" "<<counter[2]<<" Rounds, Which Is Good For "

<<fixed<<setprecision(2)<<showpoint<<pPerc<<"% Of The Rounds Played.\n";

out<<name[3]<<" "<<counter[3]<<" Rounds, Which Is Good For "

<<fixed<<setprecision(2)<<showpoint<<cPerc<<"% Of The Rounds Played.\n";

//Exit stage right!

in.close();

out.close();

return 0;

}

//Function To Either Start The Game Or Exit

bool isTrue(char choice){

cout<<endl<<endl;

cin>>choice;

cout<<endl;

if(choice!='Y' && choice!='y'){//Validating The User Input

cout<<"Exiting Game.\n";

exit(EXIT\_FAILURE);

}else{

return true;

}

}

//Function To Start and Loop Game

int whoWon(int pRank, int pSuit, int cRank, int cSuit, vector<string> name, int counter[]){

int a, b, c, d;

a=b=c=d=0;

string cnt;//Variable For Ternary Operator

do{

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

cnt=(counter[0]>0 && counter[1]>0)?"Continue The Game, Flip A Card.\n":"Game Over.\n";

cout<<cnt;

}while(counter[0]>0 && counter[1]>0);

}

void flipCard(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name){

srand(static\_cast<unsigned int>(time(0)));//Used For Random Number Generator

char choice;

cout<<endl;

cout<<"To Flip The Card, Press Y.\n";

cin>>choice;

cout<<endl;

validate(choice);//Validates User Input

//Set Card Values And Suits

pRank=rand()%13+1;

pSuit=rand()%4+1;

cRank=rand()%13+1;

cSuit=rand()%4+1;

cards(pRank, pSuit, cRank, cSuit, name);

}

//Function For Validating User Input

void validate(char choice){

while(choice!='Y' && choice!='y'){

cout<<"Invalid Entry, Please Press Y To Flip The Card.\n";

cin>>choice;

cout<<endl;

}

}

//Outputs Card Value And Suit

void cards(int pRank, int pSuit, int cRank, int cSuit, vector<string> name){

cout<<name[0]<<" Played ";

switch(pRank){

case 1:cout<<"Ace ";break;

case 2:cout<<"Two ";break;

case 3:cout<<"Three ";break;

case 4:cout<<"Four ";break;

case 5:cout<<"Five ";break;

case 6:cout<<"Six ";break;

case 7:cout<<"Seven ";break;

case 8:cout<<"Eight ";break;

case 9:cout<<"Nine ";break;

case 10:cout<<"Ten ";break;

case 11:cout<<"Jack ";break;

case 12:cout<<"Queen ";break;

case 13:cout<<"King ";break;

}

switch(pSuit){

case 1:cout<<"Of Clubs.\n";break;

case 2:cout<<"Of Diamonds.\n";break;

case 3:cout<<"Of Hearts.\n";break;

case 4:cout<<"Of Spades.\n";break;

}

cout<<name[1]<<" Played ";

switch(cRank){

case 1:cout<<"Ace ";break;

case 2:cout<<"Two ";break;

case 3:cout<<"Three ";break;

case 4:cout<<"Four ";break;

case 5:cout<<"Five ";break;

case 6:cout<<"Six ";break;

case 7:cout<<"Seven ";break;

case 8:cout<<"Eight ";break;

case 9:cout<<"Nine ";break;

case 10:cout<<"Ten ";break;

case 11:cout<<"Jack ";break;

case 12:cout<<"Queen ";break;

case 13:cout<<"King ";break;

}

switch(cSuit){

case 1:cout<<"Of Clubs.\n";break;

case 2:cout<<"Of Diamonds.\n";break;

case 3:cout<<"Of Hearts.\n";break;

case 4:cout<<"Of Spades.\n";break;

}

cout<<endl;

}

//Outputs If Player 1 Wins And Keeps Counters Updated

void whoWon(int pRank, int cRank, vector<string> name, int counter[], int a){

if(pRank>cRank){

counter[0]--;

counter[1]--;

a+=2;

cout<<name[2]<<" This Round!\n"<<endl;

//Counters

counter[2]++;

counter[3]+=0;

counter[0]+=a;

counter[1]+=0;

cout<<name[0]<<" Has "<<counter[0]<<" Cards.\n";

cout<<name[1]<<" Has "<<counter[1]<<" Cards.\n";

cout<<endl;

}

}

//Outputs If Player 2 Wins And Keeps Counters Updated

void lesser(int pRank, int cRank, vector<string> name, int counter[], int b){

if(pRank<cRank){

counter[0]--;

counter[1]--;

b+=2;

cout<<name[3]<<" This Round!\n"<<endl;

//Counters

counter[2]+=0;

counter[3]++;

counter[0]+=0;

counter[1]+=b;

cout<<name[0]<<" Has "<<counter[0]<<" Cards.\n";

cout<<name[1]<<" Has "<<counter[1]<<" Cards.\n";

cout<<endl;

}

}

//Functions For When Card Values Are Equal, And Has Each Possible Case

int equal(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank){

counter[0]--;

counter[1]--;

equal1(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal2(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal3(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal4(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal5(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal6(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal7(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal8(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal9(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal10(pRank, pSuit, cRank, cSuit, name, counter, a, b);

equal11(pRank, pSuit, cRank, cSuit, name, counter, a, b);

}

}

//Equal Case And Keeps Counter Going

int equal1(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[],int a, int b){

if(pRank==cRank && counter[0]>4 && counter[1]>4){

cout<<"It's War!\n";

cout<<"Both Place 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

counter[1]-=3;

a+=8, b+=8;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal2(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[0]==4){

cout<<"It's War!\n";

cout<<"Both Place 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

counter[1]-=3;

a+=8, b+=8;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal3(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[1]==4){

cout<<"It's War!\n";

cout<<"Both Place 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

counter[1]-=3;

a+=8, b+=8;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal4(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[0]==3){

cout<<"It's War!\n";

cout<<name[0]<<" Can Only Put 2 Cards Down, "<<name[1]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=2;

counter[1]-=3;

a+=7, b+=7;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal5(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a , int b){

if(pRank==cRank && counter[1]==3){

cout<<"It's War!\n";

cout<<name[1]<<" Can Only Put 2 Cards Down, "<<name[0]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

counter[1]-=2;

a+=7, b+=7;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal6(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[0]==2){

cout<<"It's War!\n";

cout<<name[0]<<" Can Only Put 1 Card Down, "<<name[1]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=1;

counter[1]-=3;

a+=6, b+=6;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal7(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[1]==2){

cout<<"It's War!\n";

cout<<name[1]<<" Can Only Put 1 Card Down, "<<name[0]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

counter[1]-=1;

a+=6, b+=6;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

equal(pRank, pSuit, cRank, cSuit, name, counter, a, b);

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal8(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[0]==1){

cout<<"It's War!\n";

cout<<name[0]<<" Cannot Put Any Cards Down, "<<name[1]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[1]-=3;

a+=5, b+=5;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

if(pRank==cRank){

counter[0]--;

flipCoin(name, counter, a, b, 0, 1);

pRank=0, cRank=1;

}

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal9(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[1]==1){

cout<<"It's War!\n";

cout<<name[1]<<" Cannot Put Any Cards Down, "<<name[0]<<" Will Put 3 Cards Face Down.\n";

cout<<"Each Flip Another Card.\n";

cout<<endl;

counter[0]-=3;

a+=5, b+=5;

flipCard(pRank, pSuit, cRank, cSuit, name);

whoWon(pRank, cRank, name, counter, a);

lesser(pRank, cRank, name, counter, b);

if(pRank==cRank){

counter[1]--;

flipCoin(name, counter, a, b, 1, 0);

pRank=0, cRank=1;

}

return 0;

}

}

//Equal Case And Keeps Counter Going

int equal10(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[0]==0){

cout<<"It's War!\n";

cout<<name[0]<<" Cannot Put Any Cards Down, "<<name[1]<<" Will Put 3 Cards Face Down.\n";

counter[1]-=3;

a+=5, b+=5;

flipCoin(name, counter, a, b, 0, 1);

}

return 0;

}

//Equal Case And Keeps Counter Going

int equal11(int &pRank, int &pSuit, int &cRank, int &cSuit, vector<string> name, int counter[], int a, int b){

if(pRank==cRank && counter[1]==0){

cout<<"It's War!\n";

cout<<name[1]<<" Cannot Put Any Cards Down, "<<name[0]<<" Will Put 3 Cards Face Down.\n";

counter[0]-=3;

a+=5, b+=5;

flipCoin(name, counter, a, b, 1, 0);

}

return 0;

}

//Function To Flip Coin For When A Player Doesn't Have Enough Cards

void flipCoin(vector<string> name, int counter[], int a, int b, int c, int d){

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

unsigned int coin;

char flip;

cout<<"Flip A Coin, Choose Heads(1) Or Tails(0).\n";

//Choose Heads Or Tails

cin>>flip;

cout<<endl;

while(flip!='1' && flip!='0'){//Validate User Input

cout<<"Invalid Entry, Please Choose Heads(1) Or Tails(0).\n";

cin>>flip;

cout<<endl;

}

//Flip Coin

coin=rand()%2;

cout<<coin<<endl<<endl;

//Output Who Wins And Keep Counters Going

if((flip=='1' && coin==1) || (flip=='0' && coin==0)){

cout<<name[2]<<" The War!\n"<<endl;

counter[2]++;

counter[3]+=0;

counter[0]+=a;

counter[1]+=0;

cout<<name[c]<<" Has "<<counter[c]<<" Cards.\n";

cout<<name[d]<<" Has "<<counter[d]<<" Cards.\n";

cout<<endl;

}

else{

cout<<name[3]<<" The War!\n"<<endl;

counter[2]+=0;

counter[3]++;

counter[0]+=0;

counter[1]+=b;

cout<<name[c]<<" Has "<<counter[c]<<" Cards.\n";

cout<<name[d]<<" Has "<<counter[d]<<" Cards.\n";

cout<<endl;

}

}