Project 1
Chopsticks

Bryanna Phan
Dr. Lehr
CSC-5 46091
07/21/15

Table of Contents

Introduction:

Welcome to the game of Chopsticks. This is a popular children's game often played during recess. Other popular names for this game are Sword, Sticks, and Magic Fingers.

Directions to Play:

To play, we first start off with both partners holding out one finger on each hand. You often pick someone to go first. In my program, the user will go first. Next, you tap one of your opponent's fingers. Your opponent must then hold out one additional finger on the hand you tapped (for a total of two) because the hand you used to tap them with had one finger held out. Let your opponent tap your hand. If they tap you with the hand that has one finger held out, you must hold out one additional finger on your hand that they tapped (totaling two). If they tap you with their hand that has two fingers held out, then you must add two fingers to your hand that they tapped (totaling three). Keep taking turns tapping hands and adding fingers, but when a hand has five fingers held out, that is called a "dead hand". Put dead hands behind the player's back. The person who reaches two dead hands first loses.

Project Summary:

- **Lines of Code:** ~245 lines
- **Number of Variables:**
  - char - 3
  - short - 4
  - int - 1
  - float - 1
  - ofstream - 1
- **Number of Functions: 2**

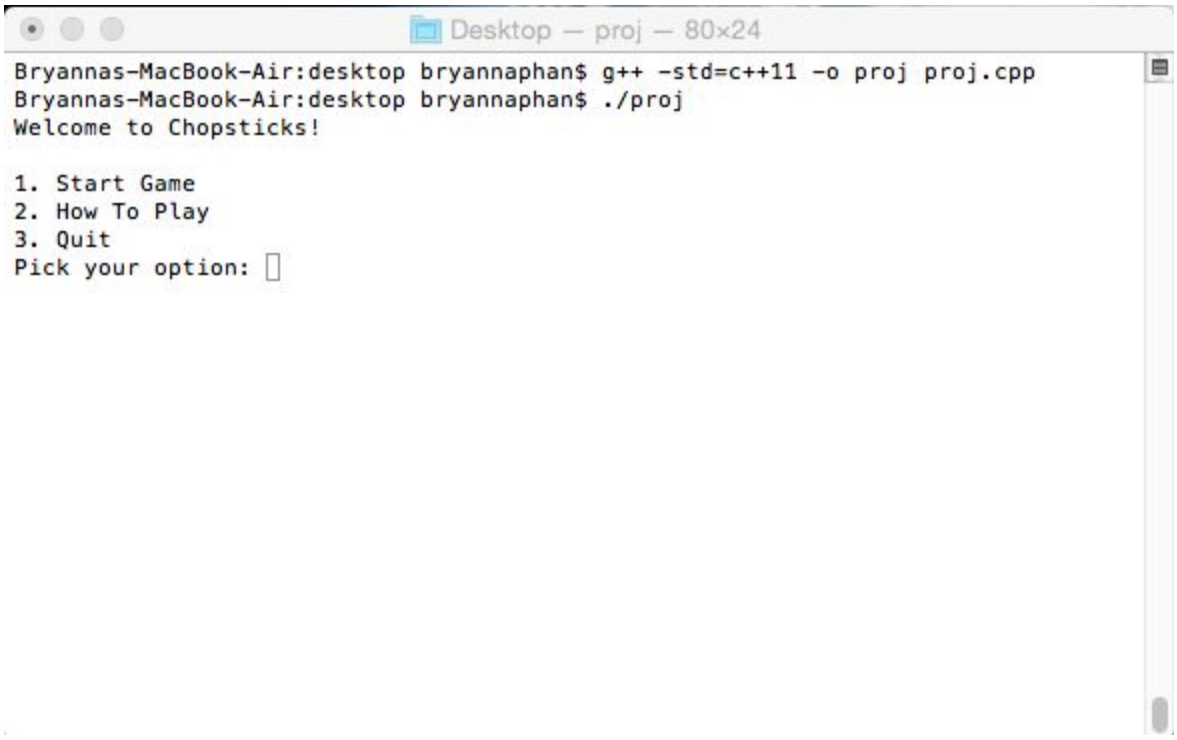
Thoughts:

I believe that I chose a very good game for our level of C++. However, there are still many limitations with this program. Although I was able to utilize many of concepts we have covered in class so far, my code includes many if-else statements which I believe can be more efficiently coded using arrays to account for all of the "hands" (the player's hands and the computer's hands). Because there were 4 hands to account for, my code turned out a lot longer than expected.

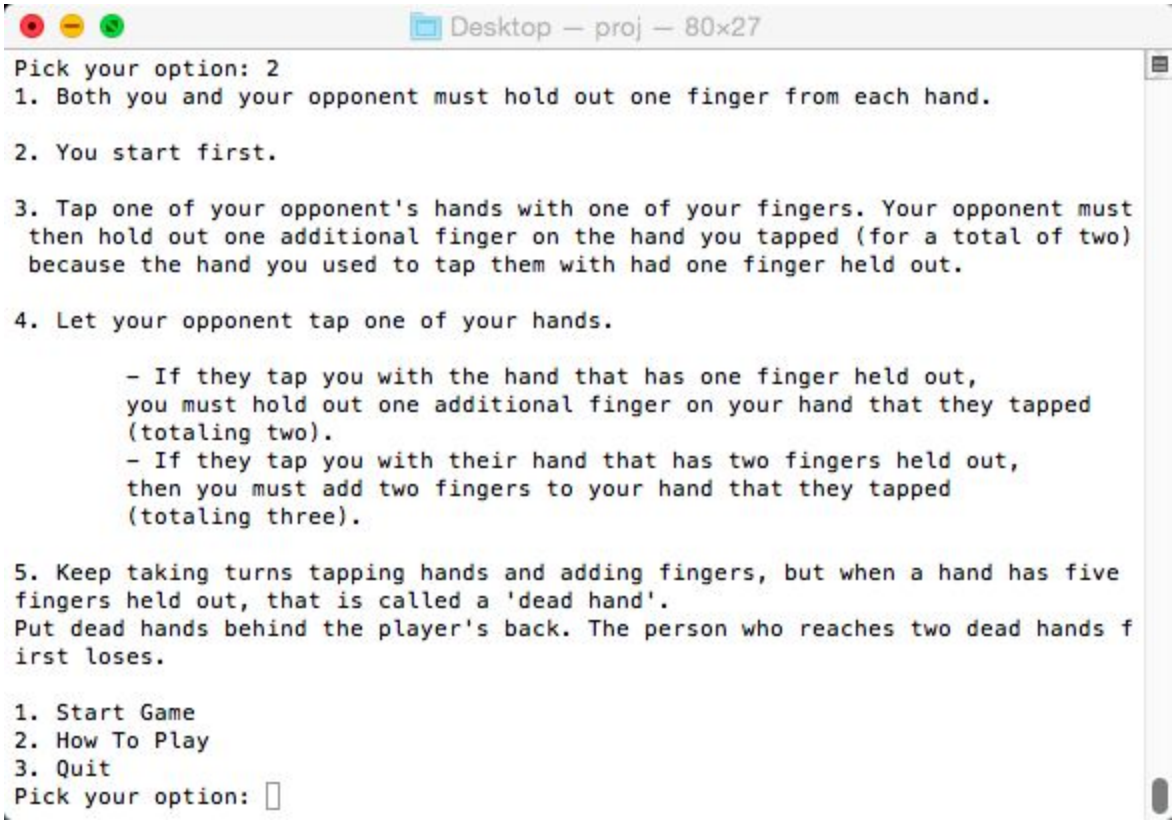
There are also some bugs with the "dead" statements. I intend to fix those problems once I implement arrays in Project 2. Other concepts I implemented into my program are a switch statement, i/o files, void functions, for-loops, while loops, if-else statements, nested loops, cin/cout statements, different variables, "tolower" function, and different orders of operations using the (+=) shortcuts we learned.

Running the Program

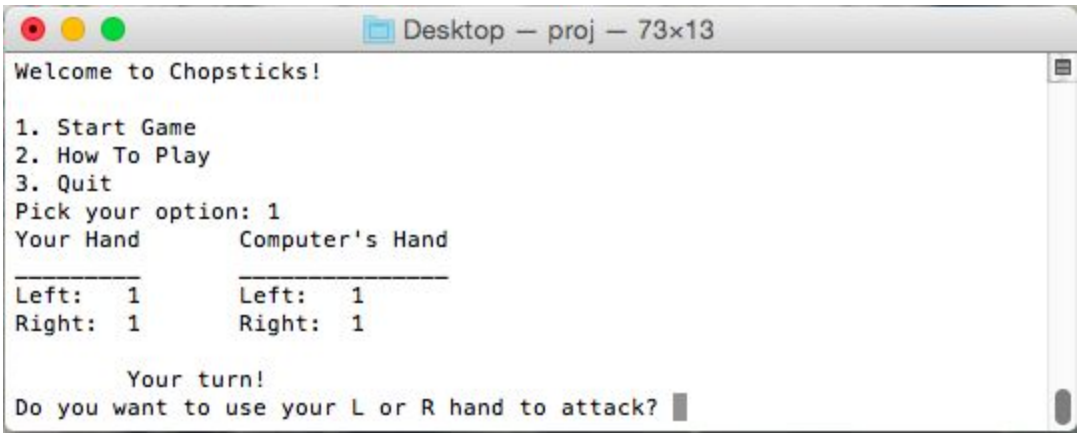
1. **Output Menu**
   - User has a choice to pick whether to start game, read instructions or quit



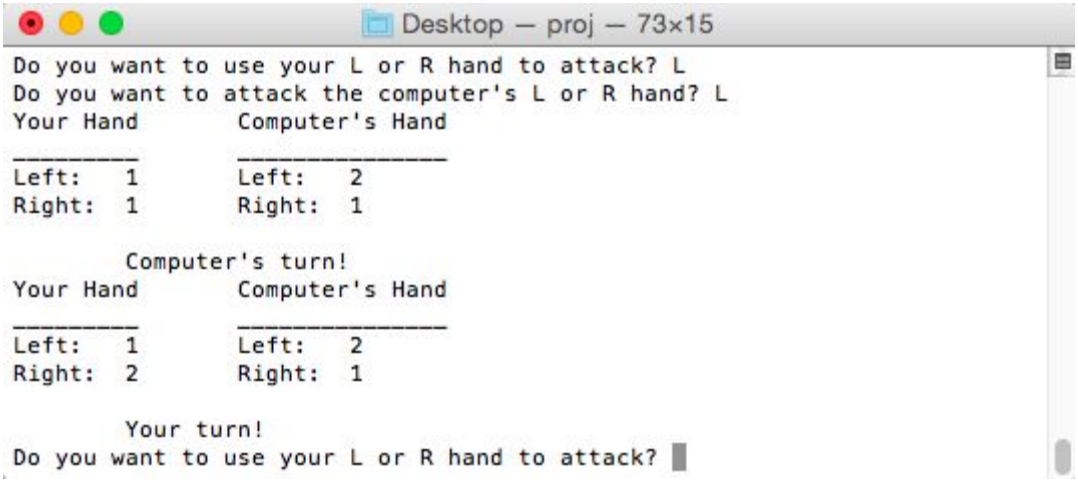
2. To view instructions
   - The function prints out the instructions and then reprints the menu



3. Starting the Game



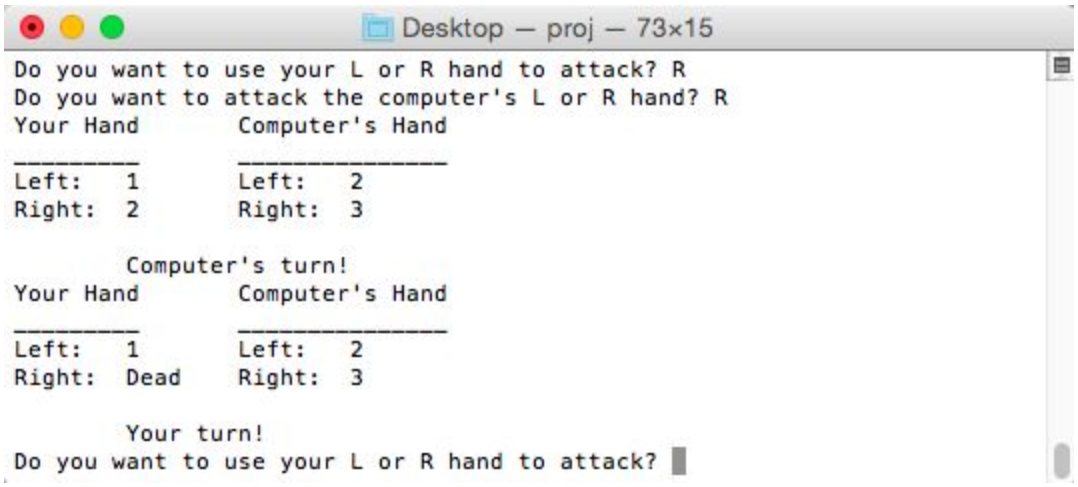
- It is the player's turn first. Input left (L/l) or right (R/r) to pick which hand to use to attack the computer.



- After picking a hand to use to attack and which computer hand TO attack, the program should output an updated chart of how many fingers each person has.

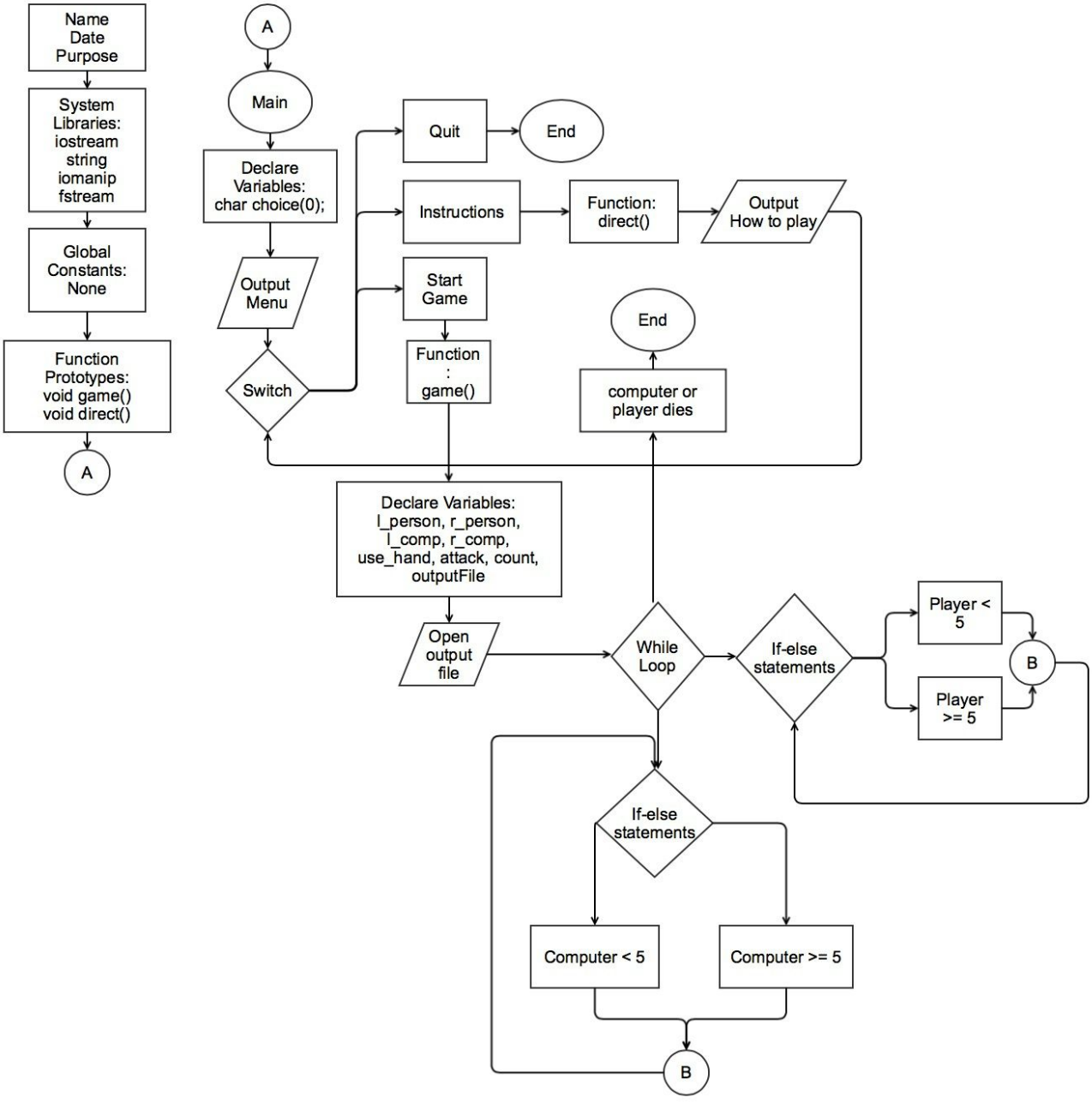
4. What is “Dead”?

- When one hand has 5 or more fingers on it, it is considered “dead”



- This means that the player’s right hand is “dead” and can no longer be used to attack the computer

Flowchart


Name
Date
Purpose
System
Libraries:
iostream
string
iomanip
fstream
Global
Constants:
None
Function
Prototypes:
void game()
void direct()
A
A
Main
Declare
Variables:
char choice(0);
Output
Menu
Switch
Quit
End
Instructions
Function:
direct()
Output
How to play
Start
Game
Function
:
game()
End
computer or
player dies
Declare Variables:
l_person, r_person,
l_comp, r_comp,
use_hand, attack, count,
outputFile
Open
output
file
While
Loop
If-else
statements
Player <
5
Player
>= 5
B
If-else
statements
Computer < 5
Computer >= 5
B

Pseudo-Code

System libraries: iostream, string, iomanip, fstream
User libraries
Global constants
Function prototypes: void game(); void direct();
I/O stream namespace

Main
- *Declare Variables: choice*
- *Menu*
- *Switch Statement (choice)*
    1. *Game*
    2. *Directions*
    3. *Quit*

End main

Direction function
- *Output directions to play*
- *End*

Game function
- *Declare variables: short l_person, r_person, l_comp, r_ comp;*
  char use_hand, attack;
  float count;
  ofstream outputFile;
- *Open file*
- *Display initial amount of fingers each player starts with*
- *While loop to end when one user wins*
    - *For loop to count how many moves*
    - *count ++ & output total moves*
- *If & if-else statement to output "win" or "lose"*
- *Output statement to show how many fingers each person has*
- *If-else statements to show which hands are "dead"*
- *Get input from user*
- *Output error if user inputs a "dead" hand*
- *Outputs updated amount of fingers*
- *Computer's turn*
- *If-else & else-if statements for computer's moves*
- *Close output file*
- *End function*

Major Variables

Type	**Variable Name**	**Description/Function**	**Location/Line** (1st occurrence)
char	choice	used for the menu in the switch statement	25
	use_hand	the hand that the player wants to use (L/R)	88, 157
	attack	the hand the player uses to attack (L/R)	89, 160
short	l_person	amount of fingers in person's left hand	86
	r_person	amount of fingers in person's right hand	86
	l_comp	amount of fingers in comp's left hand	87
	r_comp	amount of fingers in comp's right hand	87
float	count	used to count how many moves to win	95, 107
ofstream	outputFile	records how many moves to win	91, 93, 243

C++ Concepts

Chapter	Concept	Syntax/Keywords	Location
2	Display Output: cout	cout <<	32

	Assigning Statements	short l_person(1), r_person(1);
		short l_comp(1), r_comp(1);
		
		char use_hand, choice(1), attack(1);
		
		ofstream outputFile;
			86

	Arithmetic Operators	l_comp += l_person;	176
3	Read Input: cin	cin >> use_hand;	157
	Evaluating Mathematical Expressions	l_comp += l_person;	176
	Combined Assignment operators	l_comp += l_person;	176

4	if/else Statement	if (l_person >= 5)
		cout << "Dead" << '\t';
		else
		cout << l_person << '\t';
			127-130

	if/else-if Statement	if (l_person >= 5 && r_person >= 5) else if (l_comp >= 5 && r_comp >= 5)	111 - 116
	Switch statements	switch(choice)	38-60
5	while Loop	while (r_person <= 5 && l_person <=5 && r_comp <= 5 && l_comp <= 5)	103
	for Loop	for (int i=0; i<5; i++)	105
6	Defining and Calling Functions	game(); direct();	42, 48
	ofstream Output File	outputFile.open("moves.txt", ios::out | ios::app);	93

Program Code

```
/*
 * File:   proj.cpp
 * Author: Bryanna Phan
 * Purpose: Play chopsticks with the computer
 * PROJECT #1
 * Created on July 14, 2015, 12:14 AM
 */

// System Libraries
#include <iostream>
#include <string>
#include <iomanip>
#include <fstream>

// User Libraries
// Global Constants
// Function Prototypes
void game(); // void function for the game
void direct(); // void function to output instructions on how to play

using namespace std;

int main() {
        // Declare Variables
        char choice(0); // used for switch statement

        // introduction
        cout << "Welcome to Chopsticks!" << endl << endl;

        while (choice != 3)
        {
                cout << "1. Start Game" << endl;
                cout << "2. How To Play" << endl;
                cout << "3. Quit" << endl;
                cout << "Pick your option: ";
                cin >> choice;;

                switch(choice) // switch statement for the menu
                {
                        case '1':
```

```
                {
                        game();
                        break;
                }

                case '2':
                {
                        direct();
                        break;
                }

                case '3':
                {
                        cout << "The game has ended." << endl;
                        return 0;
                        break;
                }

                default: cout << "You didn't enter an option between 1-3. Please try
again." << endl;
                }
        }
return 0;
}


void direct() { // output directions to play

        cout << "1. Both you and your opponent must hold out one finger from each hand." <<
endl << endl;
        cout << "2. You start first." << endl << endl;
        cout << "3. Tap one of your opponent's hands with one of your fingers. Your opponent
must then hold out one additional finger on the hand you tapped (for a total of two) because the
hand you used to tap them with had one finger held out." << endl << endl;
        cout << "4. Let your opponent tap one of your hands." << endl << endl;
        cout << "       - If they tap you with the hand that has one finger held out," << endl;
        cout << "       you must hold out one additional finger on your hand that they tapped" <<
endl;
        cout << "       (totaling two)." << endl;
        cout << "       - If they tap you with their hand that has two fingers held out," << endl;
        cout << "       then you must add two fingers to your hand that they tapped" << endl;
        cout << "       (totaling three)." << endl << endl;
```

```
	cout << "5. Keep taking turns tapping hands and adding fingers, but when a hand has five fingers held out, that is called a 'dead hand'." << endl;
	cout << "Put dead hands behind the player's back. The person who reaches two dead hands first loses." << endl << endl;
}

void game() // function for the game
{
	short l_person(1), r_person(1), // player's left and right hand
	l_comp(1), r_comp(1);			// computer's left and right hand
	char use_hand,		// which hand the user is using to attack computer
	attack(1);				// which computer hand user wants to attack
	float count(0);

	ofstream outputFile;
	// open input file to show how many moves each game was
	outputFile.open("moves.txt", ios::out | ios::app);

	// display INITIAL AMT of fingers each player has
	cout << "Your Hand" << '\t' << "Computer's Hand" << endl;
	cout << "__________" << '\t' << "__________________" << endl;
	cout << "Left:" << '\t' << l_person << '\t' << "Left:" << '\t' << l_comp << endl;
	cout << "Right:" << '\t' << r_person << '\t' << "Right:" << '\t' << r_comp << endl << endl;

	while (r_person <= 5 && l_person <=5 && r_comp <= 5 && l_comp <= 5)
	{
		for (int i=0; i<5; i++) // count how many moves
		{
			count++;
			outputFile << "Total moves: " << count << endl;
		}

		if (l_person >= 5 && r_person >= 5)
		{
			cout << "You lose." << endl;
		}

		else if (l_comp >= 5 && r_comp >= 5)
		{
			cout << "You win!." << endl;
		}
```

```
// display amount of fingers each player has
cout << "Your Hand" << '\t' << "Computer's Hand" << endl;
cout << "__________" << '\t' << "__________________" << endl;
cout << "Left:" << '\t';

// output statements to show which hands are "dead"
if (l_person >= 5)
        cout << "Dead" << '\t';
else
        cout << l_person << '\t';

cout << "Left:" << '\t';

if (l_comp >= 5)
        cout << "Dead" << endl;
else
        cout << l_comp << endl;

cout << "Right:" << '\t';

if (r_person >= 5)
        cout << "Dead" << '\t';
else
        cout << r_person << '\t';

cout << "Right:" << '\t';

if (r_comp >= 5)
        cout << "Dead" << endl;
else
        cout << r_comp << endl << endl;;

// user's turn
cout << '\t' << "Your turn!" << '\t' << endl;

// user input
cout << "Do you want to use your L or R hand to attack? ";
cin >> use_hand;
use_hand = tolower(use_hand);
cout << "Do you want to attack the computer's L or R hand? ";
cin >> attack;
attack = tolower(attack);
```

```
		// determining person's moves
		if (use_hand == 'l' && l_person >= 5)
		{
			while (use_hand == 'l')
			{
				cout << "That hand is dead. Please choose another." << endl;
				cin >> use_hand;
				use_hand = tolower(use_hand);
			}
		}
		else if (use_hand == 'l' && l_person < 5)
		{
			if (attack == 'l')
				l_comp += l_person;
			else if (attack == 'r')
				r_comp += l_person;
		}

		if (use_hand == 'r' && r_person >= 5)
		{
			while (use_hand == 'r')
			{
				cout << "That hand is dead. Please choose another." << endl;
				cin >> use_hand;
				use_hand = tolower(use_hand);
			}
		}
		else if (use_hand == 'r' && r_person < 5)
		{
			if (attack == 'l')
				l_comp += r_person;
			else if (attack == 'r')
				r_comp += r_person;
		}

		// display amount of fingers each player has
		cout << "Your Hand" << '\t' << "Computer's Hand" << endl;
		cout << "__________" << '\t' << "__________________" << endl;
		cout << "Left:" << '\t' << l_person << '\t' << "Left:" << '\t' << l_comp << endl;
		cout << "Right:" << '\t' << r_person << '\t' << "Right:" << '\t' << r_comp << endl <<
endl;
		cout << '\t' << "Computer's turn!" << '\t' << endl;
```

```
            if (r_comp >= 3 && r_comp < 5)
            {
                  if (r_person >= 2 && r_person <= 5)
                        r_person += r_comp;
                  else if (l_person >= 2 && l_person <= 5)
                        l_person += r_comp;
            }

            else if (l_comp >= 3 && l_comp < 5)
            {
                  if (r_person >= 2 && r_person <= 5)
                        r_person += l_comp;
                  else if (l_person >= 2 && r_person <= 5)
                        l_person += l_comp;
            }

            else if (l_comp == 1 && r_person == 1 && l_person == 1)
            {
                  if (r_person == 1)
                        r_person += l_comp;
                  else if (l_person == 1)
                        l_person += l_comp;
            }

            else if (r_comp == 1 && r_person == 1 && l_person == 1)
            {
                  if (r_person == 1)
                        r_person += r_comp;
                  else if (l_person == 1)
                        l_person += r_comp;
            }

            else if (l_comp > 1 && l_comp < 5)
                  r_person += l_comp;
            else if (r_comp > 1 && r_comp < 5)
                  l_person += r_comp;
      }

outputFile.close();
}
```