

Game Documentation

by Alex Archer

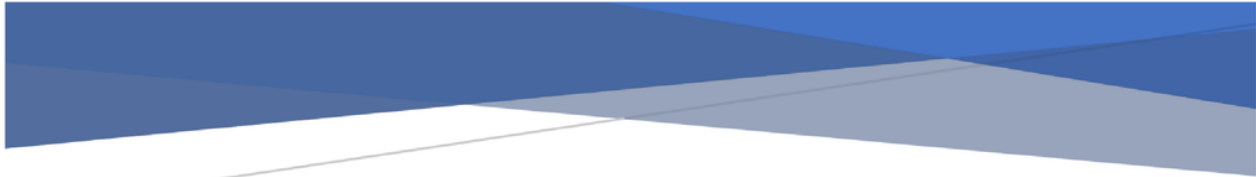
Submission date: 05-Apr-2019 05:41PM (UTC+0100)

Submission ID: 104229120

File name: 1637343341.Game_Documentation.docx (12.91M)

Word count: 2112

Character count: 11626



SOFTWARE DEVELOPMENT

Alex Archer

152940

Applied Computing

Due: May 9th 2019

Contents

Higher or Lower	2
Introduction	2
Stages of Development	3
Problems Solved	8
Testing	10
Evaluation	12
Treasure Hunt	13
Introduction	13
Stages of Development	14
Testing	15
Evaluation	16
Bibliography	17
Appendix	18
Higher or Lower Code	18
Treasure Hunt Code	20

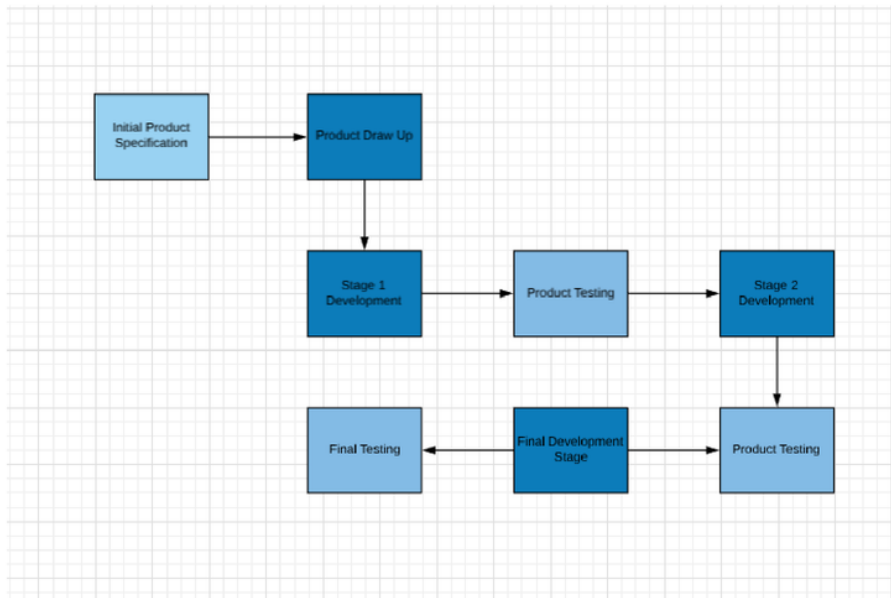
Higher or Lower

Introduction

To begin this assignment, I was tasked with designing a higher or lower game which provided players with short term fun. Before starting the development process of the game, I had to understand the challenges and problems a developer could face. To ensure I gave myself enough time to prepare, I did as much research on C++ as I could, to reduce the number of bugs I ran into. I purchased a book which would provide me with enough information about this new programming language (Stroustrup, 2013). I also engaged in an online course using CodeCademy to put my newly learnt skills to the test (Zach Sims, 2019). I had a decent enough understanding about C++ so I could start thinking about the development of the first game, Higher or Lower.

Stages of Development

Before development began, I had to create a product development flow chart which allowed me to visualise the process of each stage, which is a critical part of product development in the industry. I demonstrated my flow chart using different colours to make it clearer to read for the user/client. The flow chart was a very useful tool to refer back to whenever I felt I had lost track of the process of my first game using C++.



As you can see from the flow chart - I created, in my head, an initial product specification which meant that I could visualize the outcome of the game and what features I will have to include to ensure the development would be successful. I ensured I included at least 2 testing stages before a final test to finalize the product. With some of the problems a new programmer could run into, I had a few syntax errors which were easily fixed once I had increased my understanding of the language.

```

miscellaneous files
1 // higherOrLower.cpp : This file contains the 'main' function. Program execution begins and ends there.
2 //
3
4 #include "pch.h"
5 #include <iostream>
6 #include <string>
7 using namespace std;
8
9 void enterNumbers();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Higher or Lower!\n" << endl;
13     cout << "Player 1 enter 7 numbers\n\n";
14     enterNumbers();
15
16     return 0;
17 }
18

```

To begin, I created a simple output which included a welcome message to the game. I used an include at the start of my .cpp document which meant that I didn't have to include std:: before every input/output declaration. This made my code look a lot cleaner and easier to type as a developer. To do this, I used "using namespace std;".

I included a placeholder for the function I was going to call from later in my code, this meant that even if I wanted to use the function again in the future, it was available to me.

```

1 // higherOrLower.cpp : This file contains the 'main' function. Program execution begins and ends there.
2 //
3
4 #include "pch.h"
5 #include <iostream>
6 #include <string>
7 using namespace std;
8
9 void enterNumbers();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Higher or Lower!\n" << endl;
13     cout << "Player 1 enter 7 numbers\n\n";
14     enterNumbers();
15
16     return 0;
17 }
18
19 //<--Player 1 section begins-->
20 void enterNumbers() {
21     int z, i;
22     int temp = 0;
23     int playerOneChoices[7] = { 0,0,0,0,0,0,0 };
24
25     for (z = 1; z < 8; z++) {
26         cout << "Enter number " << z << ": "; //Player one chooses 7 numbers (Does not include 1-10 range)
27         cin >> playerOneChoices[z]; //Stores values into the array, until z reaches 7
28     }
29
30     for (i = 0; i < 7; i++) {
31         if (playerOneChoices[i] > temp)
32             temp = playerOneChoices[i];
33     }
34 }
35

```

To then further my code, I initialised an array which would accept 7 numbers from the user (player one) and created a loop which had a counter that would reach 8 before breaking from the loop. The loop included the array inputs. I created a comment which would allow me to come back to the code and understand any problems I would need to fix at the end of my development process.

```

8
9 void enterNumbers();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Higher or Lower!\n" << endl;
13     cout << "Player 1 enter 7 numbers\n\n";
14     enterNumbers();
15
16     return 0;
17 }
18
19 //<--Player 1 section begins-->
20 void enterNumbers() {
21     int z, i;
22     int temp = 0;
23     int playerOneChoices[7] = { 0,0,0,0,0,0,0 };
24
25     for (z = 1; z < 8; z++) {
26         cout << "Enter number " << z << ": "; //Player one chooses 7 numbers (Does not include 1-10 range)
27         cin >> playerOneChoices[z]; //Stores values into the array, until z reaches 7
28     }
29
30     for (i = 0; i < 7; i++) {
31         if (playerOneChoices[i] > temp)
32             temp = playerOneChoices[i];
33     }
34
35     system("CLS"); //Clears screen so player 2 can't see player 1 choices
36
37     //<--Player 2 section begins-->
38     bool correct = true;
39     int counter = 1;
40     string guess;
41
42     cout << "\nPlayer 2, the first number is " << playerOneChoices[1] << "\n";
43     //Player 2 guesses Higher or Lower
44
45
46
47

```

Once player one's numbers were inputted, it was time to make a start on player two's guesses. I initialised a Boolean variable which was set to true, this would be in case player two guesses wrong, which would set the Boolean to false, resulting in player one winning the game. To further evaluate the code written, the use of << allowed me to include the array numbers in a cout, to ensure player two knew which number was being guessed higher or lower. I also included system("CLS");

So that player two had no idea what numbers were inputted by player one. I believe this was the most useful line of code throughout the entire game as without it would make the game virtually unplayable.

```

32     if (playerOneChoices[i] > temp)
33         temp = playerOneChoices[i];
34     }
35
36     system("CLS"); //Clears screen so player 2 can't see player 1 choices
37
38     //<--Player 2 section begins-->
39     bool correct = true;
40     int counter = 1;
41     string guess;
42
43     cout << "\nPlayer 2, the first number is " << playerOneChoices[1] << "\n";
44     //Player 2 guesses Higher or Lower
45
46     while ((counter < 7) && (correct == true)) {
47         cout << "Higher (H) or Lower (L)? ";
48         cin >> guess;
49         if (guess == "H") { //If guess is "H", execute the next line
50             if (playerOneChoices[counter + 1] > playerOneChoices[counter]) { //If next number is more than the previous, then proceed with the next number
51                 cout << "Correct, the next number is " << playerOneChoices[counter + 1] << endl;
52             }
53             else {
54                 cout << endl << "Wrong, Player One wins!" << endl; //If number is lower than the previous, player 2 loses
55                 correct = false;
56             }
57             counter++; //Repeat loop until all 6 numbers have been guessed
58         }
59         else if (guess == "L") {
60             if (playerOneChoices[counter + 1] < playerOneChoices[counter]) {
61                 cout << "Correct, the next number is " << playerOneChoices[counter + 1] << endl;
62             }
63             else {
64                 cout << endl << "Wrong, Player One wins!" << endl;
65                 correct = false;
66             }
67             counter++;
68         }
69         else {
70             cout << "Enter either H or L" << endl;
71         }
72     }
73
74

```

To understand my thought process when going into this section, I basically created a loop which had a value which counted to 7 and if the counter reached 7, player 2 would win. To ensure this code ran smoothly, I went through thorough testing to make sure bugs were resolved. The code is easy enough to read, it allows the user to input either H or L, if the next number was higher or lower than the previous, then the game would either end or commence with the next number, until counter reaches 7.

```

65     correct = false;
66 }
67 counter++;
68 }
69 else {
70     cout << "Enter either H or L" << endl;
71 }
72
73
74 //<---Player 2 wins section--->
75 if (correct == true) {
76     cout << endl << endl << "The last number was " << playerOneChoices[7] << "!" << endl << "----Congratulations! Player Two wins!----" << endl;
77 }
78
79
80
81
82
83

```

I then created a section at the end for player 2 winning the game. This was just a combination of the variable staying true and outputs stating that the player has won. This was the end of the code used for the higher or lower game. Here is an image of the code in full:

```

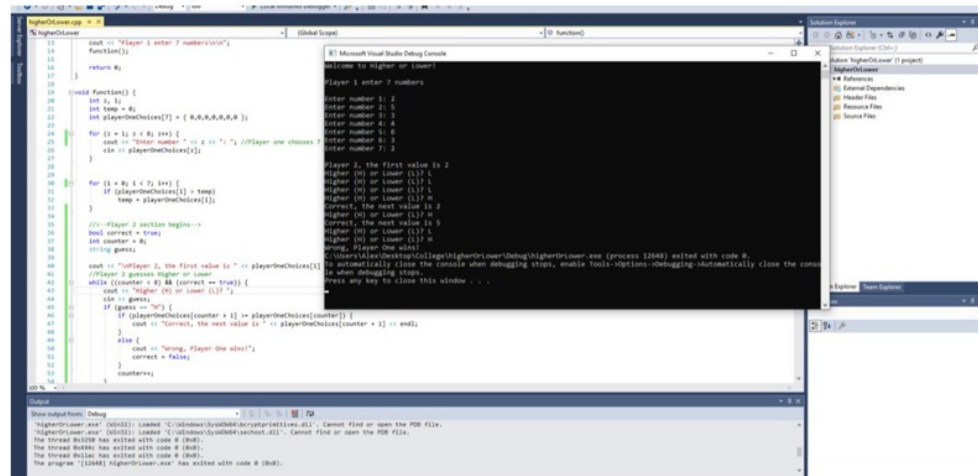
1 // higherOrLower.cpp : This file contains the 'main' function. Program execution begins and ends there.
2 //
3
4 #include "pch.h"
5 #include <iostream>
6 #include <string>
7 using namespace std;
8
9 void enterNumbers();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Higher or Lower!\n" << endl;
13     cout << "Player 1 enter 7 numbers!\n\n";
14     enterNumbers();
15
16     return 0;
17 }
18
19 //Player 1 section begins--
20 void enterNumbers() {
21     int z, i;
22     int temp = 0;
23     int playerOneChoices[7] = { 0,0,0,0,0,0,0 };
24
25     for (z = 1; z <= 8; z++) {
26         cout << "Enter number " << z << " "; //Player one chooses 7 numbers (Does not include 1-30 range)
27         cin >> playerOneChoices[z]; //Stores values into the array, until z reaches 7
28     }
29
30     for (i = 0; i < 7; i++) {
31         if (playerOneChoices[i] > temp)
32             temp = playerOneChoices[i];
33     }
34
35     system("CLS"); //Cleans screen so player 2 can't see player 1 choices
36
37     //Player 2 section begins--
38     bool correct = true;
39     int counter = 1;
40     string guess;
41
42     cout << "Player 2, the first number is " << playerOneChoices[1] << "\n";
43     //Player 2 guesses Higher or Lower
44
45     while ((counter < 7) && (correct == true)) {
46         cout << "Higher (H) or Lower (L)? ";
47         cin >> guess;
48         if (guess == "H") { //If guess is "H", execute the next line
49             if (playerOneChoices[counter + 1] > playerOneChoices[counter]) { //If next number is more than the previous, then proceed with the next number
50                 cout << "Correct, the next number is " << playerOneChoices[counter + 1] << endl;
51             }
52             else {
53                 cout << endl << "Wrong, Player One wins!" << endl; //If number is lower than the previous, player 2 loses
54                 correct = false;
55             }
56             counter++; //Repeat loop until all 8 numbers have been guessed
57         }
58         else if (guess == "L") {
59             if (playerOneChoices[counter + 1] < playerOneChoices[counter]) {
60                 cout << "Correct, the next number is " << playerOneChoices[counter + 1] << endl;
61             }
62             else {
63                 cout << endl << "Wrong, Player One wins!" << endl;
64                 correct = false;
65             }
66             counter++;
67         }
68         else {
69             cout << "Enter either H or L" << endl;
70         }
71     }
72
73     //Player 2 wins section----
74     if (correct == true) {
75         cout << endl << endl << "The last number was " << playerOneChoices[7] << "!" << endl << "----Congratulations! Player Two wins!----" << endl;
76     }
77 }

```

As you can see from the line numbers on the left, the entire game was only 77 lines of code. I believe this would be one of the more efficient ways of creating the higher or lower game. The code does have a few bugs, however. I will talk more about that in the next section.

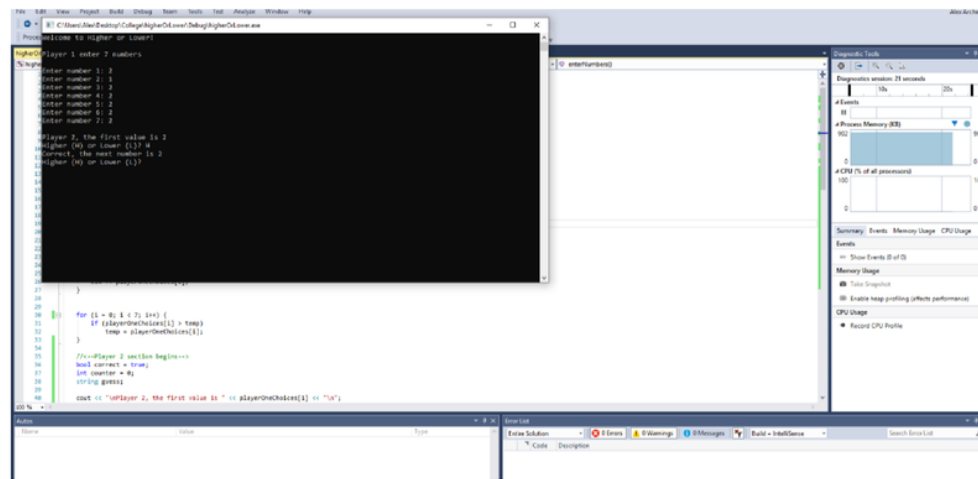
While developing the higher or lower game, I ran into a few bugs which either set the code into an infinite loop, or meant that the game had a slightly different outcome to what I was going for. I compiled a few screenshots which visualize the bugs through the windows console. Here they are, easiest to hardest:

Problem 1) Easiest



I ran into an issue that outputs the H or L question twice, before the game begins. Also, if the guess was "L", it would infinitely repeat the question, until "H" was typed. To fix this, I included an else if statement for if guess was assigned "L". This fixed the guess = "L" issue.

Problem 2) Moderate

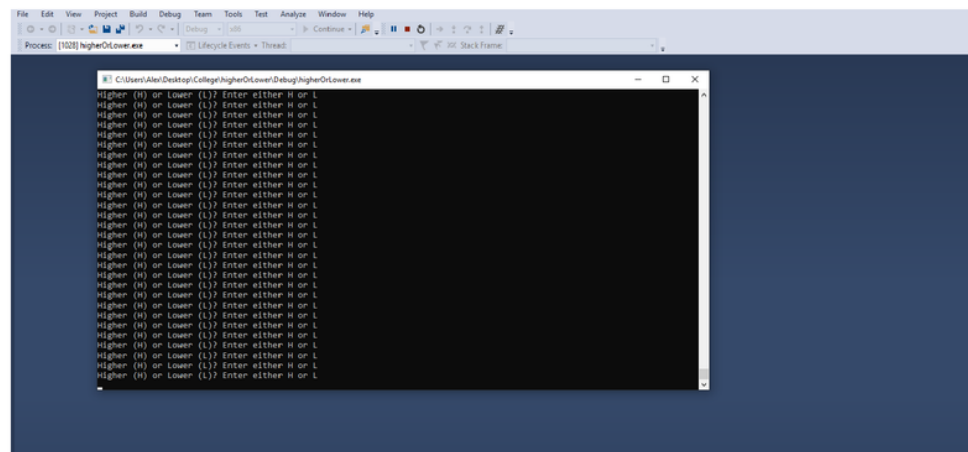


I was still stuck with repeating value 1 before the game began. To fix this issue, I had to change the line:

```
int counter = 1;
```

The counter began at 0, resulting in the array pull to count null, meaning it would repeat the first number twice.

Problem 3) Hardest



The third problem, which has been noticed but not resolved, is that the game goes into an infinite loop if player 1 enters anything other than an integer. I understand that this is a simple fix. However, the way I've written my code means that I would have to potentially re-write the entire player one section.

Testing

Test Number	What am I testing?	Expected Outcome	Evidence
1	Inputting numbers for player ones selection: 5, 3, 2, 1, 5, 6, 7	Array to hold numbers 5, 3, 2, 1, 5, 6, 7 and then first number to output for player two guesses	 
2	Testing "H" works for player two guess	Correct or Incorrect answer, outputs next number	
3	Next value should have the same input option of "H" or "L"	Correct or Incorrect answer, outputs next number	
4	Player two guesses incorrect, player one wins	Game ends and outputs player one is the winner	

Evaluation

To conclude the ¹ higher or lower game, I believe the game was a success. However, even though I didn't resolve one of the bugs, the game is still viable to play between two users. The process of development for the game has allowed me to improve my coding and development skills, while also giving me valuable skills when following a process for product development with a client.

Treasure Hunt

Introduction

To begin the second game of this assignment, I was tasked with creating a treasure hunt game using multidimensional arrays to output a grid layout. The game itself is played with two players and the winner is declared if player two finds all three treasures placed by player one without running out of lives.

Stages of Development

Using the same strategy from game one, I used the same flowchart to ensure my work was completed in the most efficient and bug-free way possible.

```
1 // Treasure Hunt.cpp : This file contains the 'main' function. Program execution begins and ends there.
2 //
3
4 #include "pch.h"
5 #include <iostream>
6 #include <string>
7 using namespace std;
8
9 void enterCoord();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Treasure Hunt!\n" << endl;
13     cout << "Player One place treasures now:\n\n";
14     enterCoord();
15
16     return 0;
17 }
18
```

I used the same format for the welcome message and initial instruction as the higher or lower game. Once again, I used a placeholder function to call from later on in my document.

```
8
9 void enterCoord();
10 //Welcome message
11 int main() {
12     cout << "Welcome to Treasure Hunt!\n" << endl;
13     cout << "Player One place treasures now:\n\n";
14     enterCoord();
15
16     return 0;
17 }
18
19 }
20
21 //---Player 1 section begins---
22 void enterCoord() {
23     int z, i;
24     int temp = 0;
25     int playerOneChoices[6] = { 0,0,0,0,0,0 };
26     string gridLayout[5][5];
27     gridLayout = { " ", " ", " ", " ", " " }, z;
28
29     for (z = 1; z < 4; z++) {
30         cout << "X: "; //Player one chooses 6 X & Y coordinates
31         cin >> playerOneChoices[z]; //Stores values into the array, until z reaches 6
32         cout << "Y: ";
33         cin >> playerOneChoices[z];
34         cout << "Treasure " << z << " placed\n";
35     }
36 }
37
38
39
```

Moving along, I created a for loop which asked the user to input a set of X and Y coordinated 3 times. Once a set of coordinates were declared, the system would output the treasure number and that it was placed. I ran into an issue which meant that I couldn't initialise a multi-dimensional array to form the grid. This was the biggest issue in my code, unfortunately resulting in no further progress.

Testing

Test Number	What am I testing?	Expected Outcome	Evidence
1	Correct number of coordinates and array saved	System asked player one 3 times for coordinates and then stores them into the array	

Evaluation

To evaluate the treasure hunt game, I was very upset that I couldn't overcome the array issue. I believe that the game would have been playable if I tackled that issue. However, to say I failed the task is probably the wrong thing to say. I consider this a learning curve for a new developer like myself.

Bibliography

Stroustrup, B. (2013). *The C++ Programming Language* (4th Edition ed.). Boston: Addison Wesley.

Zach Sims, R. B. (2019). Retrieved from CodeCademy: <https://www.codecademy.com>

Appendix

Higher or Lower Code

```
// higherOrLower.cpp : This file contains the 'main' function. Program execution
// begins and ends there.
//

#include "pch.h"
#include <iostream>
#include <string>
using namespace std;

void enterNumbers();
//Welcome message
int main() {
    cout << "Welcome to Higher or Lower!\n" << endl;
    cout << "Player 1 enter 7 numbers\n\n";
    enterNumbers();

    return 0;
}
//<--Player 1 section begins-->
void enterNumbers() {
    int z, i;
    int temp = 0;
    int playerOneChoices[7] = { 0,0,0,0,0,0,0 };

    for (z = 1; z < 8; z++) {
        cout << "Enter number " << z << ": "; //Player one chooses 7 numbers
        (Does not include 1-10 range)
        cin >> playerOneChoices[z]; //Stores values into the array, until z
        reaches 7
    }

    for (i = 0; i < 7; i++) {
        if (playerOneChoices[i] > temp)
            temp = playerOneChoices[i];
    }
    system("CLS"); //Clears screen so player 2 can't see player 1 choices

    //<--Player 2 section begins-->
    bool correct = true;
    int counter = 1;
    string guess;

    cout << "\nPlayer 2, the first number is " << playerOneChoices[1] << "\n";
    //Player 2 guesses Higher or Lower

    while ((counter < 7) && (correct == true)) {
        cout << "Higher (H) or Lower (L)? ";
        cin >> guess;
        if (guess == "H") { //If guess is "H", execute the next line
            if (playerOneChoices[counter + 1] >= playerOneChoices[counter]) {
                //If next number is more than the previous, then proceed with the next number
                cout << "Correct, the next number is " <<
                playerOneChoices[counter + 1] << endl;
            }
            else {

```

```

        cout << endl << "Wrong, Player One wins!" << endl; //If
number is lower than the previous, player 2 loses
        correct = false;
    }
    counter++; //Repeat loop until all 6 numbers have been guessed
}
else if (guess == "L") {
    if (playerOneChoices[counter + 1] <= playerOneChoices[counter]) {
        cout << "Correct, the next number is " <<
playerOneChoices[counter + 1] << endl;
    }
    else {
        cout << endl << "Wrong, Player One wins!" << endl;
        correct = false;
    }
    counter++;
}
else {
    cout << "Enter either H or L" << endl;
}
}
//<----Player 2 wins section---->
if (correct == true) {
    cout << endl << endl << "                The last number was " <<
playerOneChoices[7] << "!" << endl << "<----Congratulations! Player Two wins!---->" <<
endl;
}
}

```

Treasure Hunt Code

```
// Treasure Hunt.cpp : This file contains the 'main' function. Program execution
// begins and ends there.
//

#include "pch.h"
#include <iostream>
#include <string>
using namespace std;

void enterCoord();
//Welcome message
int main() {
    cout << "Welcome to Treasure Hunt!\n" << endl;
    cout << "Player One place treasures now:\n\n";
    enterCoord();

    return 0;

}
//<--Player 1 section begins-->
void enterCoord() {
    int z, i;
    int temp = 0;
    int playerOneChoices[6] = { 0,0,0,0,0,0 };
    string gridLayout[5][5];
    gridLayout = { '-', '-', '-', '-', '-' }, ;

    for (z = 1; z < 4; z++) {
        cout << "X: "; //Player one chooses 6 X & Y coordinates
        cin >> playerOneChoices[z]; //Stores values into the array, until z
reaches 6
        cout << "Y: ";
        cin >> playerOneChoices[z];
        cout << "Treasure " << z << " placed\n";
    }

}
```

Game Documentation

ORIGINALITY REPORT

7%

SIMILARITY INDEX

3%

INTERNET SOURCES

0%

PUBLICATIONS

7%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to Pembrokeshire College

Student Paper

4%

2

Submitted to Glasgow Clyde College

Student Paper

2%

3

journals.plos.org

Internet Source

1%

4

Submitted to University of Lincoln

Student Paper

<1%

Exclude quotes Off

Exclude bibliography Off

Exclude matches Off

Game Documentation

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

40/100

Instructor

Exercise 1 33/50%

You have demonstrated how you will break down the problem with a block diagram. Flowchart and/or pseudo code is useful to break down the problem further and aid implementation.

I would recommend separating the two main features of the function `enterNumbers()` into two separate functions, the first to collect the player one values and the second to collect and check player two's guesses. The main function then would just handle the logic of reviewing the guesses and deciding which player has won. Your use of functions simply shunts the contents of the main program out into a function, this implies you have not really understood the use of a function in programming. The original brief specified player one was only to enter values between 1 and 10, this is neither implemented or tested, having done so would have given you further marks.

Exercise 2 7/30%

As you have only implemented capturing and storing values for this program, and as a result you have limited your marks in this section. You could have ensured values were inputted between 0 and 7 which were the ranges specified for the treasure grid to gain further marks. You could also have attempted to implement this as a one dimensional array using the code suggested in the assignment brief.

Exercise 3 0/ 20%

You have not attempted this question

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21
