

proframming

by Ruzy Aktar

FILE	7200_RUZY_AKTAR_PROFRAMMING_178138_264318614.DOCX (577.28K)		
TIME SUBMITTED	06-APR-2019 12:00AM (UTC+0100)	WORD COUNT	1548
SUBMISSION ID	104255031	CHARACTER COUNT	7670



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Software Development

Programming in C

Mike Dacey/ Tim Bashford

Ruzy Aktar | 1708215 | 5/04/2019

Introduction

In programming it is very important to execute code systematically, the student will be understanding the syntax and semantics of high-level programming language and develop programs by executing appropriate practices. Also, the student will be using simple basic algorithms to develop solutions to figure out any programming difficulties. Along with solving the problems this student will be making a list of problems solved easiest to hardest. Evidence of problems being solved before coding, this can be through drawings, written notes etc. however, also stages of the development such as distinguishing the patterns applied through each step and the associated tests. As well as that the student will be providing screenshots of results, with description of the aim of the test, the input and the expected outputs.

EXERCISE 1

For the first exercise the student will be building a two-player game, the game works by the first player entering 7 values between 1 and 10. The first value is shown to the second player in the game. Both players must figure out if the next value is higher or lower than the one being shown. After entering their guess, the next value is presented out and this process will repeat until all 7 values are shown.

However, after all values are shown the game will end and the second player wins, furthermore if the second person inputs incorrect guessed then first player will win. Here is an example of the game and what it should look like. Below the student will be now trying this using visual studio and providing own screenshots of the process of building this game. This student will be using arrays to help and functions to create the game, below is an example of what the game should appear to be like.

```
Player 1 please enter 7 values:
Enter number 1 > 2
Enter number 2 > 4
Enter number 3 > 8
Enter number 4 > 3
Enter number 5 > 9
Enter number 6 > 5
Enter number 7 > 1
} The first player enters these values

Player 2 the first value is 2
Higher (H) or lower (L) > h
Correct, next value is 4
Higher (H) or lower (L) > h
Correct, next value is 8
Higher (H) or lower (L) > l
Correct, next value is 3
Higher (H) or lower (L) > l
Incorrect, sorry player 1 wins
```

```

#include <string>
#include <iostream>
#include <sstream>

using namespace std;

int ReadNumber(string message);
void ReadSevenNumbers(int numbers[]);
string ReadHL();
bool CheckHL(int count, string guess, int numbers[]);

int main(int argc, char*argv[])
{
    int playerOne[6];
    string playerTwo;
    int count;
    string input;
    stringstream stream;

```

So the student to start off the game has created pieces of code. The top lines are function declarations which tells the compiler that the functions exist within the program.

The following lines after the main function are the variables that the student has created. Variables which an item that can increase more than one value during its runtime in the program.

```

cout << "Player 1 please enter 7 values (1-10) \n" << endl;
ReadSevenNumbers(playerOne);

cout << "\nPlayer 2 you must guess Player 1 values, first value is given. \n" << endl;

for (count = 1; count < 7; count++)
{
    cout << "The value is " << playerOne[count - 1] << endl;

    playerTwo = ReadHL();
    if (CheckHL(count, playerTwo, playerOne) == false)
    {
        break;
    }
}

if (count == 7)
{
    cout << "Player 2 wins " << endl;
}

cin.get();
return 0;

```

Here above in this screenshot it shows the next step the student has created to make the game. Here the student has added a print to data to console, this acts as a friendly user interface, this is important when considering the user experience. This the first line and second line in the screenshot. The rest of the code the student has created a For Loop. A for loop is important and is acquainted to monitor for certain conditions and then constantly execute blocks from codes to those conditions met. Inside the For Loop it's checking if the player two has guessed the right value of player one's input. If not, it will break

To check if player two guesses are equal to the seven values of player one, then this will display if player two wins. This is the If statement shown on the screenshot above

And this is the outcome, it shows the player one has entered 7 values to start off the game. Within the function contains a For Loop which allows player one to enter a value 7 times. This is linked to the input text pattern within the Main. And below shows the student has been able to do this successfully. Also, the while Loop makes sure the player cannot enter anything above 10 must stay between 1-10.

```
void ReadSevenNumbers(int numbers[])
{
    for (int index = 0; index < 7; index++)
    {
        numbers[index] = ReadNumber("Enter number: ");
        while (numbers[index] < 1 || numbers[index] > 10) {
            cout << "Values must be between 1 and 10" << endl;
            index -= 1;
            break;
        }
    }
}
```

```
cout << "Player 1 please enter 7 values (1-10) \n" << endl;
ReadSevenNumbers(playerOne);
```

```
Player 1 please enter 7 values (1-10)
Enter number: 7
Enter number: 4
Enter number: 2
Enter number: 6
Enter number: 5
Enter number: 8
Enter number: 2
Player 2 you must guess Player 1 values, first value is given.
```

So as you can see here the student has been able to build the first part of the program. The game is asking the player one to enter 7 values using the first input text pattern, and as you can see the player has done this. and now the second input text pattern is put into place and is asking player two to guess player ones values . In order for these values

```

bool CheckHL(int count, string guess, int numbers[])
{
    if (guess == "H")
    {
        if (numbers[count] >= numbers[count - 1])
        {
            cout << "correct" << endl;
            return true;
        }
        else
        {
            cout << "wrong guess player 1 wins" << endl;
            return false;
        }
    }
    else // "L"
    {
        if (numbers[count] < numbers[count - 1])
        {
            cout << "correct" << endl;
            return true;
        }
        else
        {
            cout << "wrong guess player 1 wins" << endl;
            return false;
        }
    }
}

```

```

Enter number: 7
Enter number: 4
Enter number: 2
Enter number: 6
Enter number: 5
Enter number: 8
Enter number: 2

Player 2 you must guess Player 1 values, first value is given.

The value is 7
Enter H or L:L
correct
The value is 4
Enter H or L:L
correct
The value is 2
Enter H or L:H
correct
The value is 6
Enter H or L:L
correct
The value is 5
Enter H or L:H
correct
The value is 8
Enter H or L:L
correct
Player 2 wins

```

To complete the last stages of the game, player two must now guess player one's values. Above in the report is a screenshot of player one inserting 7 values on page 3. As you can see above player two has won by making the right guess correctly throughout the game.

The student was able to accomplish this by using a set of functions such as Bool. This function withholds two values to determine if it's true or false. So now the if statement it checks if the guess of player two (H or L) is equal to more than or less than the values of player one. If so, the bool will check this, if the assumption matches player one's values then the next value of player one will be shown in order, this process will repeat until player two has correctly guessed all six values of player one. Determining player two to become the winner, or until player two makes an incorrect guess resulting in player one becoming the winner.

Below is a screenshot of showing what it would look like if player two loses.

```
Player 1 please enter 7 values (1-10)
Enter number: 7
Enter number: 4
Enter number: 2
Enter number: 6
Enter number: 5
Enter number: 8
Enter number: 2

Player 2 you must guess Player 1 values, first value is given.
The value is 7
Enter H or L:H
wrong guess player 1 wins
```

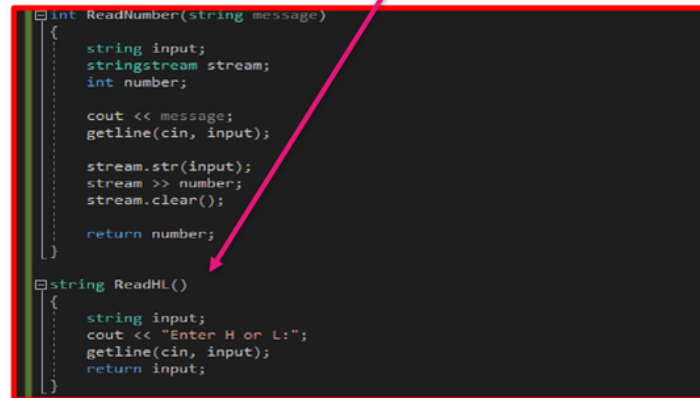
Also, the while Loop previously mentioned this is an example of what it would look like if player one was to enter a number above 10.

```
void ReadSevenNumbers(int numbers[])
{
    for (int index = 0; index < 7; index++)
    {
        numbers[index] = ReadNumber("Enter number: ");
        while (numbers[index] < 1 || numbers[index] > 10) {
            cout << "Values must be between 1 and 10" << endl;
            index -= 1;
            break;
        }
    }
}
```



```
Player 1 please enter 7 values (1-10)
Enter number: 7
Enter number: 4
Enter number: 12
Values must be between 1 and 10
Enter number:
```

And lastly below in the screen shot the first half of it is a number pattern, which allows player one to enter the numbers. And then the 'string ReadhL' code is to allow the player two to guess higher or lower.



```
int ReadNumber(string message)
{
    string input;
    stringstream stream;
    int number;

    cout << message;
    getline(cin, input);

    stream.str(input);
    stream >> number;
    stream.clear();

    return number;
}

string ReadhL()
{
    string input;
    cout << "Enter H or L:";
    getline(cin, input);
    return input;
}
```

Exercise2

Build a two player game where the first player hides 3 hidden treasures and the second has to guess where they are by entering the X and Y coordinates of a location on an 8x8 grid. The player starts with 3 lives. If they fail to find the treasure then an N character should be placed on the grid and they lose a life. An X should be displayed if they find the treasure, and they win if they discover all three treasures. The player grid should be re-displayed after each guess, as well as appropriate messages such as "You found the first treasure". The game might look like this:

```
Player 1 place treasures now:
X: 1
Y: 1
Treasure 1 placed
X: 4
Y: 5
Treasure 2 placed
X: 7
Y: 6
Treasure 3 placed
  0  1  2  3  4  5  6  7
0  -  -  -  -  -  -  -  -
1  -  -  -  -  -  -  -  -
2  -  -  -  -  -  -  -  -
3  -  -  -  -  -  -  -  -
4  -  -  -  -  -  -  -  -
5  -  -  -  -  -  -  -  -
6  -  -  -  -  -  -  -  -
7  -  -  -  -  -  -  -  -
Player 2 Guess
X: 1
Y: 2
You found the first treasure
  0  1  2  3  4  5  6  7
0  -  -  -  -  -  -  -  -
1  -  X  -  -  -  -  -  -
2  -  -  -  -  -  -  -  -
3  -  -  -  -  -  -  -  -
4  -  -  -  -  -  -  -  -
5  -  -  -  -  -  -  -  -
6  -  -  -  -  -  -  -  -
7  -  -  -  -  -  -  -  -
Player 2 Guess
X: 5
Y: 6
No treasure here, you have 2 lives left
You found the first treasure
  0  1  2  3  4  5  6  7
0  -  -  -  -  -  -  -  -
1  -  X  -  -  -  -  -  -
2  -  -  -  -  -  -  -  -
3  -  -  -  -  -  -  -  -
4  -  -  -  -  -  -  -  -
```

The first player enters this information

Found treasure

For this section of the assignment the student has attempted but hasn't been able to complete it. Below is what the student has done.

The student has made .

```
#include<string>
#include<iostream>
#include<sstream>

using namespace std;
int playerOne;

void PrintGrid(char grid[9][9])
{
    for (int i = 0; i < 9; i++)
    {
        for (int j = 0; j < 9; j++)
        {
            cout << grid[i][j] << " ";
        }
        cout << endl << endl;
    }
}
```

```

int main(int argc, char *argv[]) {

    string input;
    stringstream stream;
    int x;
    int y;

    char board[9][9] = {
        {'-', '0', '1', '2', '3', '4', '5', '6', '7'},
        {'0', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'1', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'2', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'3', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'4', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'5', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'6', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'7', '-', '-', '-', '-', '-', '-', '-', '-'}
    };
}

```

```
cout << " Player 1 place treasures now: " << endl;
```

```
cout << "x: ";  
getline(cin, input);  
stream.str(input);  
stream >> x;  
stream.clear();
```

```
cout << "y:";  
getline(cin, input);  
stream.str(input);  
stream >> y;  
stream.clear();  
board[x][y] = 'T';
```

```
cout << " Treasure 1 placed: " << endl;  
board[x][y] = 'T';
```

```
cout << "x: ";  
getline(cin, input);  
stream.str(input);  
stream >> x;
```

```
cout << "y:";
getline(cin, input);
stream.str(input);
stream >> y;
stream.clear();
board[x][y] = 'T';

cout << " Treasure 2 placed: " << endl;

cout << "x: ";
getline(cin, input);
stream.str(input);
stream >> x;
stream.clear();

cout << "y:";
getline(cin, input);
stream.str(input);
stream >> y;
stream.clear();
board[x][y] = 'T';

cout << " Treasure 3 placed: " << endl;
PrintGrid(board);
```

the outcome of the program .

```
Player 1 place treasures now:
x: 7
y:6
Treasure 1 placed:
x: 4
y:3
Treasure 2 placed:
x: 2
y:4
Treasure 3 placed:
- 0 1 2 3 4 5 6 7
0 - - - - - - - -
1 - - - T - - - -
2 - - - - - - - -
3 - - T - - - - -
4 - - - - - - - -
5 - - - - - - - -
6 - - - - - T - -
7 - - - - - - - -
```

Appendix exercise1.

```
#include <string>
#include <iostream>
#include <sstream>

using namespace std;

int ReadNumber(string message);
void ReadSevenNumbers(int numbers[]);
string ReadHL();
bool CheckHL(int count, string guess, int numbers[]);

int main(int argc, char*argv[])
{

    int playerOne[7];
    string playerTwo;
    int count;
    string input;
    stringstream stream;

    cout << "Player 1 please enter 7 values (1-10) \n" << endl;
    ReadSevenNumbers(playerOne);

    cout << "\nPlayer 2 you must guess Player 1 values, first value is given.
\n" << endl;

    for (count = 1; count < 7; count++)
    {
        cout << "The value is " << playerOne[count - 1] << endl;

        playerTwo = ReadHL();
        if (CheckHL(count, playerTwo, playerOne) == false)
        {
            break;
        }
    }

    if (count == 7)
    {
        cout << "Player 2 wins " << endl;
    }

    cin.get();
    return 0;
}

int ReadNumber(string message)
```

```

{
    string input;
    stringstream stream;
    int number;

    cout << message;
    getline(cin, input);

    stream.str(input);
    stream >> number;
    stream.clear();

    return number;
}

string ReadHL()
{
    string input;
    cout << "Enter H or L:";
    getline(cin, input);
    return input;
}

void ReadSevenNumbers(int numbers[])
{
    for (int index = 0; index < 7; index++)
    {
        numbers[index] = ReadNumber("Enter number: ");

        while (numbers[index] < 1 || numbers[index] > 10) {
            1 cout << "Values must be between 1 and 10" << endl;
              index -= 1;
              break;
        }
    }
}

bool CheckHL(int count, string guess, int numbers[])
{
    if (guess == "H")
    {
        if (numbers[count] >= numbers[count - 1])
        {
            cout << "correct" << endl;
            return true;
        }
        else
        {
            cout << "wrong guess player 1 wins" << endl;
            return false;
        }
    }
    else // "L"
    {
        if (numbers[count] < numbers[count - 1])
        {

```

```
        cout << "correct" << endl;
        return true;
    }
    else
    {
        cout << "wrong guess player 1 wins" << endl;
        return false;
    }
}
```

Appendix 2

```
#include<string>
#include<iostream>
#include<sstream>

using namespace std;
int playerOne;

void PrintGrid(char grid[9][9])
{
    for (int i = 0; i < 9; i++)
    {
        for (int j = 0; j < 9; j++)
        {
            cout << grid[i][j] << " ";
        }
        cout << endl << endl;
    }
}

int main(int argc, char *argv[]) {

    string input;
    stringstream stream;
    int x;
    int y;

    char board[9][9] = {
        {'-', '0', '1', '2', '3', '4', '5', '6', '7'},
        {'0', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'1', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'2', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'3', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'4', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'5', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'6', '-', '-', '-', '-', '-', '-', '-', '-'},
        {'7', '-', '-', '-', '-', '-', '-', '-', '-'}
    };

    cout << " Player 1 place treasures now: " << endl;

    cout << "x: ";
    getline(cin, input);
    stream.str(input);
    stream >> x;
    stream.clear();

    cout << "y: ";
    getline(cin, input);
    stream.str(input);
```

```

stream >> y;
stream.clear();
board[x][y] = 'T';

cout << " Treasure 1 placed: " << endl;
board[x][y] = 'T';

cout << "x: ";
getline(cin, input);
stream.str(input);
stream >> x;
stream.clear();

cout << "y: ";
getline(cin, input);
stream.str(input);
stream >> y;
stream.clear();
board[x][y] = 'T';

cout << " Treasure 2 placed: " << endl;

cout << "x: ";
getline(cin, input);
stream.str(input);
stream >> x;
stream.clear();

cout << "y: ";
getline(cin, input);
stream.str(input);
stream >> y;
stream.clear();
board[x][y] = 'T';

cout << " Treasure 3 placed: " << endl;
PrintGrid(board);

cin.get();
return 0;

```

```

}

```



FINAL GRADE

50/100

GENERAL COMMENTS

Instructor

Exercise 1 (50) 44

Exercise 2 (30) 6

Exercise 3 (20) 0

Slight errors in exercise 1, but it works and the functions are appropriate.

Exercise 2 started and some evidence that the approach be taken is correct.

I would like to see more analysis of the problems, even if this had just numbered lists of the sub-problems that needed to be addressed.

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



Comment 1

This could be an "If" statement

PAGE 16

PAGE 17

PAGE 18



Comment 2

Why are these not in a loop