Contents

Chapter 17: Problem Solving with 1-Dimensial Array – Part 3

Objectives

When we have finished this chapter, we should be able to:

1. Use 1-dimensional (1-D) array so that variables of the same purpose and type can be grouped and manipulated together.
2. Apply simple algorithm in the program
3. Test and verify the program.
4. Understand the execution of a program.

17.0 Introduction

In this chapter, we will continue to add new features into the program that developed in chapter 16 before. As a reminder ,1-dimensional array and other programming approachs like looping ,conditional statement still applied in this program. To modify this program with 1-D array we still focus on the steps, which are identifying input, process and output. In the implementation step, we will be discussed about concept of manipulate 1-D array data (basic algorithm in chapter 17) in the process stage. Finally, we need to verify the program by using different values so that the program is able to produce the expected output and it is correct.

We will follow the steps of problem solving in programming as listed below.

Step 1: Understand the Problem. Identify Input, Process and Output.

Step 3: Implement the Solution

Step 3.1: Develop the Program for a Function

Step 3.2: Test and Understand the Memory Snapshot of the Program

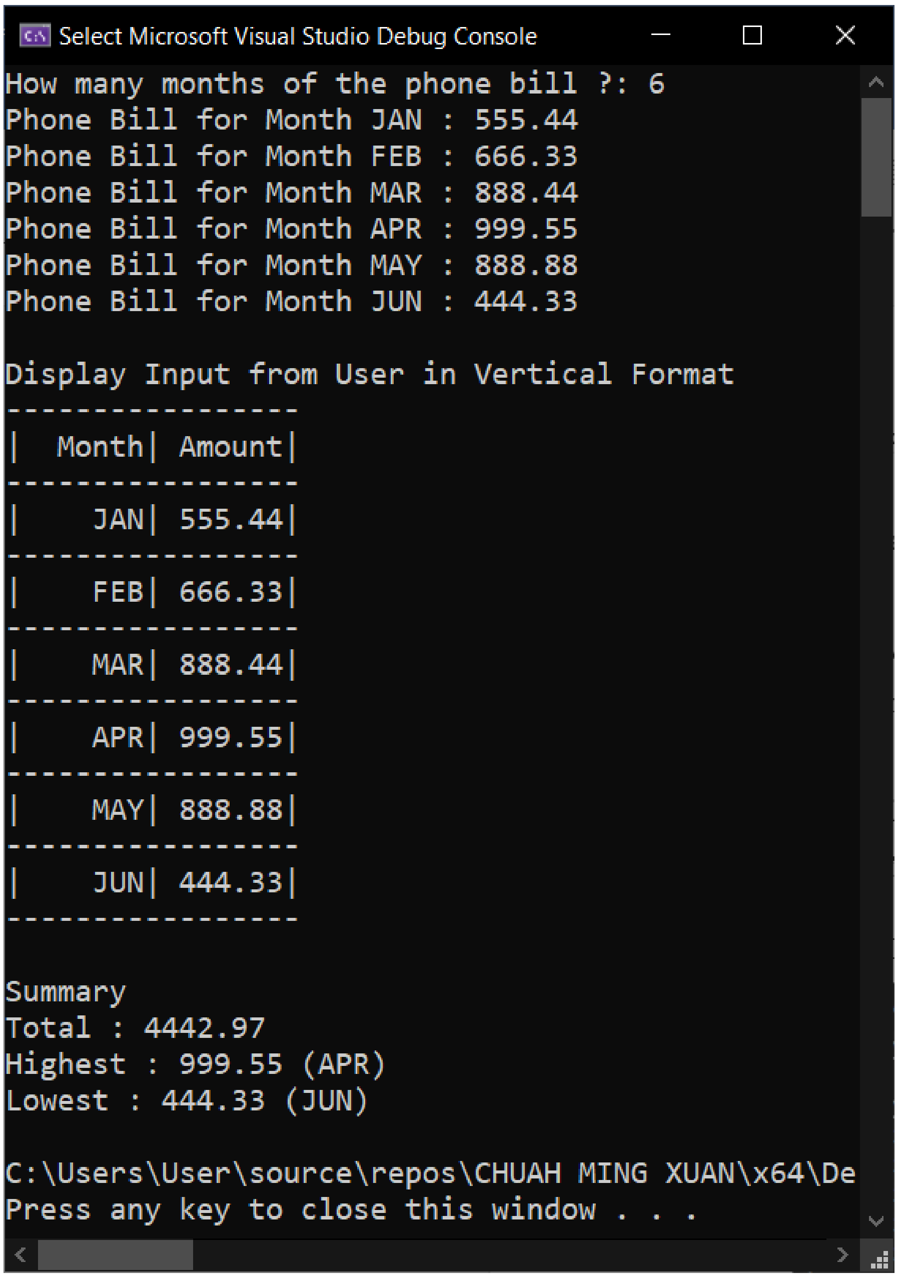
(Step 3.1 and Step 3.2 will be iterated until all functions are completed)

Step 4: Verify the Program

17.1 Develop a Program using Worked Examples

Based on Figure 17.1, we have to require user to enter how many months are needed to key in value and calculate. If the values entered is not in the range, then an error message will be displayed .After it, user input the values like before (refer back to Chapter 15 and 16) .Subsequent lines are output which is based on user inputs. The output will be displayed vertically. In addition ,a short summary that includes total ,highest bill and lowest bill will listed below the table . In this task, you are required to understand and implement a solution with **at least five (5) functions** inclusive of main function.

Example of Output Run in Console (Execution):



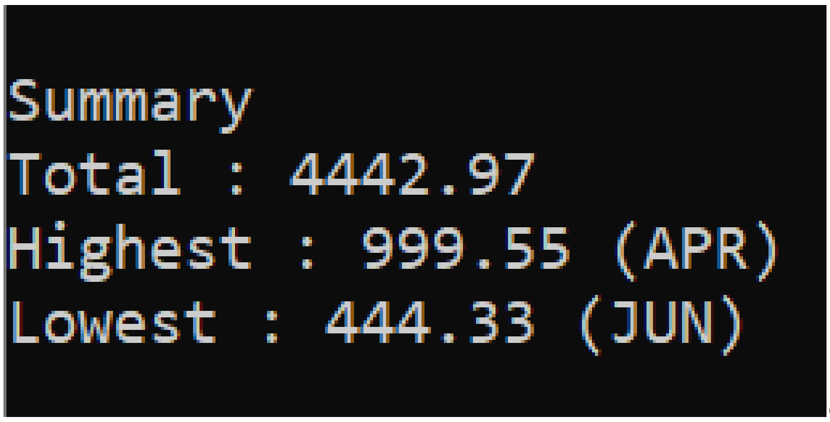
```
How many months of the phone bill ?: 6
Phone Bill for Month JAN : 555.44
Phone Bill for Month FEB : 666.33
Phone Bill for Month MAR : 888.44
Phone Bill for Month APR : 999.55
Phone Bill for Month MAY : 888.88
Phone Bill for Month JUN : 444.33

Display Input from User in Vertical Format
-----------------
|  Month| Amount|
-----------------
|    JAN| 555.44|
-----------------
|    FEB| 666.33|
-----------------
|    MAR| 888.44|
-----------------
|    APR| 999.55|
-----------------
|    MAY| 888.88|
-----------------
|    JUN| 444.33|
-----------------

Summary
Total : 4442.97
Highest : 999.55 (APR)
Lowest : 444.33 (JUN)

C:\Users\User\source\repos\CHUAH MING XUAN\x64\De
Press any key to close this window . . .
```

Figure 17. 1 Sample Input-Output of a Program - Visual Studio

Step 1: Understand the Problem

1. The main difference that can be observed from the figure 17.1 is the horizontal format is eliminated from our program. In other words ,the vertical format is chosen as the final report format in this case .Compared with vertical format table in chapter 16 ,the row of total also been removed .Although the table in figure 17.1 is similar with the vertical format table in chapter 15 which have no total column ,we still cannot copy the whole function of vertical table from chapter . The reason behind is we are required display the vertical table based on the number of months inputted. Thus ,the simplest way to reach this requirement is **we modify the function *vertical()* from chapter 16 and remove the displaying of *total row***

2. Refer to Figure 17.2 ,the total is displayed in summary. In addition, another two values are also shown in the summary which are the highest bill and the lowest bill between the months. From this information given ,we are required develop a function to find the lowest and the highest bill from the values inside array. After it ,both of these values should be returned to the main function and able to display in the output screen,



```
Summary
Total : 4442.97
Highest : 999.55 (APR)
Lowest : 444.33 (JUN)
```

of

Figure 17. 2 Understand Summary of the phone bill

3. Based on all the information that have been identified, we may create a Problem Analysis Chart (PAC) as in Figure 17.3.

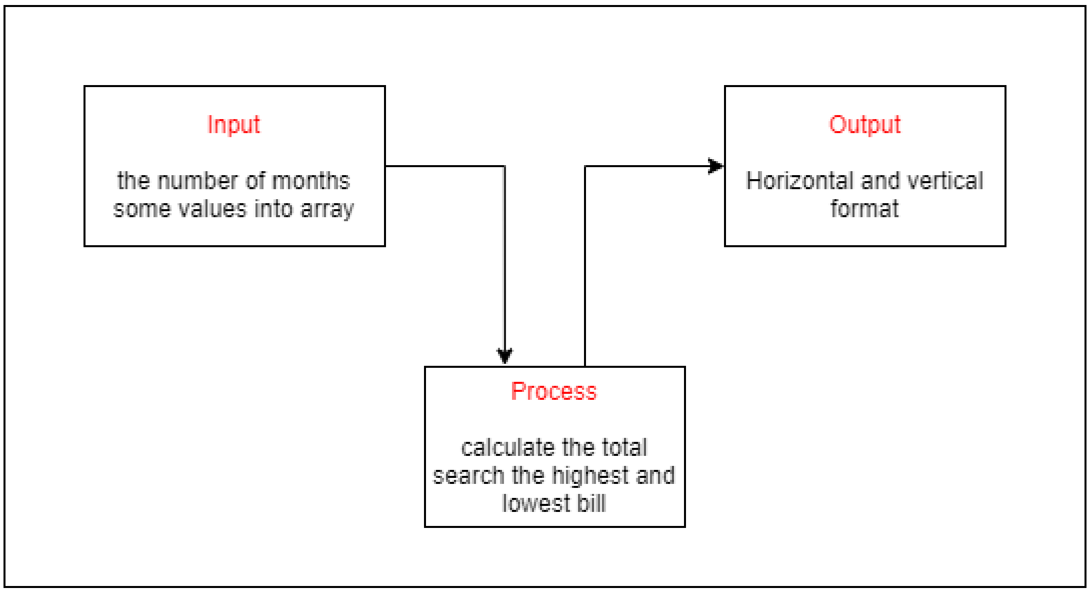


Figure 17. 3 Problem Analysis Chart (PAC) – 1-D Array

Step 2: Plan and Design Solution

For chapter 15 onwards, we assumed that students are familiar how to plan and design a solution. They are able to visualise CIPO chart while implementing a solution. We move on to Step 3.

Step 3: Implement the Solution

For the solution ,we will modify the program that already been developed in chapter 16.There are some iterations for developing additional features in the program. The iteration 1 will guide us to rewrite the function vertical() to remove the row of total. The iteration 2 will develop a function to find the highest bill and the lowest bill . The iteration 3 will develop a new function that display the summary as demonstrated in the example.

To recall back ,there is the program that been developed in the chapter 16 in the next pages. Please noted that the function horizontal was removed from the program due to this format was not selected in this case.

Table 17. 1 Previous program -part 1

Line Number	Source Code
1	`//a program that applies 1 d array`
2	`//By Chuah Ming Xuan (D032110227),FTMK ,UTeM`
3	`#include <iostream>`
4	`#include <iomanip>`
5	`using namespace std;`
6	`#define size 12 //the value of size is changed`
7	`//const int size2 = 12;`
8	
9	`int input(float Bill[], string m[])`
10	`{`
11	`        int month; //store the number of the months`
12	`        do {`
13	`                cout << "How many months of the phone bill ?: ";`
14	`                cin >> month;`
15	`                if (month > 0 && month <= size)`
16	`                        break; //exit loop when data in the range`
17	`                cout << "Invalid Input (1-12 only)" << endl;`
18	`        } while (1); //a loop always true is created`
19	`        for (int i = 0; i < month; i++)`
20	`        {`
21	`                cout << "Phone Bill for Month " << m[i] << " : ";`
22	`                cin >> Bill[i];`
23	`        }`
24	`        return month;`
25	`} //input data`
26	
27	`float calculate(float Bill[], int m)`
28	`{`
29	`        float total = 0;`
30	`        for (int i = 0; i < m; total += Bill[i++]);//calculate the total using a loop`
31	`        return total;`
32	`}`
33	

Table 17. 2 Previous program -part 2

```
Line Number | Source Code
34  void vertical(float record[], string row[], int m, float total)
35  {//line 1
36      cout << endl << "Display Input from User in Vertical Format" << endl;
37      //line 2
38      cout << setw(18) << setfill('-') << "\n";
39      //line 3 to 16
40      for (int i = -1; i <= m; i++)
41      {
42          cout << "|";
43          if (i == -1)
44          {
45            cout << setw(8) << setfill(' ') << "Month|";
46            cout << setw(8) << "Amount|";
47
48          }
49          else if (i == m)
50          {
51            cout << setw(8) << setfill(' ') << "Total|";
52            cout << setw(7) << total << "|";
53          }
54          else
55          {
56            cout << setw(7) << setfill(' ') << row[i] << "|";
57            cout << setw(7) << record[i] << "|";
58          }
59          cout << endl;
60          cout << setw(18) << setfill('-') << "\n";
61      }
62  }
63
64  int main()
65  {
66      float bill[size];
67      string monthname[] =
68  { "JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC" };//declare a array with exist values
69      int month = input(bill, monthname);
70      float sum = calculate(bill, month);
71       horizontal(bill, monthname, month, sum);
72      vertical(bill, monthname, month, sum);
73  }
```

Iterative 1 - Step 3.1: Modify Function vertical()

As mentioned before ,we have to remove the row of the total in the function `vertical()`.

Table 17. 3 Modify function vertical() –part 1(No any change)

Line Number	Source Code
1	`//a program that applies 1 d array`
2	`//By Chuah Ming Xuan (D032110227),FTMK ,UTeM`
3	`#include <iostream>`
4	`#include <iomanip>`
5	`using namespace std;`
6	`#define size 12 //the value of size is changed`
7	`//const int size2 = 12;`
8	
9	`int input(float Bill[], string m[])`
10	`{`
11	`int month; //store the number of the months`
12	`do {`
13	`cout << "How many months of the phone bill ?: ";`
14	`cin >> month;`
15	`if (month > 0 && month <= size)`
16	`break; //exit loop when data in the range`
17	`cout << "Invalid Input (1-12 only)" << endl;`
18	`} while (1); //a loop always true is created`
19	`for (int i = 0; i < month; i++)`
20	`{`
21	`cout << "Phone Bill for Month " << m[i] << " : ";`
22	`cin >> Bill[i];`
23	`}`
24	`return month;`
25	`} //input data`
26	
27	`float calculate(float Bill[], int m)`
28	`{`
29	`float total = 0;`
30	`for (int i = 0; i < m; total += Bill[i++]);//calculate the total using a loop`
31	`return total;`
32	`}`
33	

Table 17. 4 Modify function vertical() –part 2

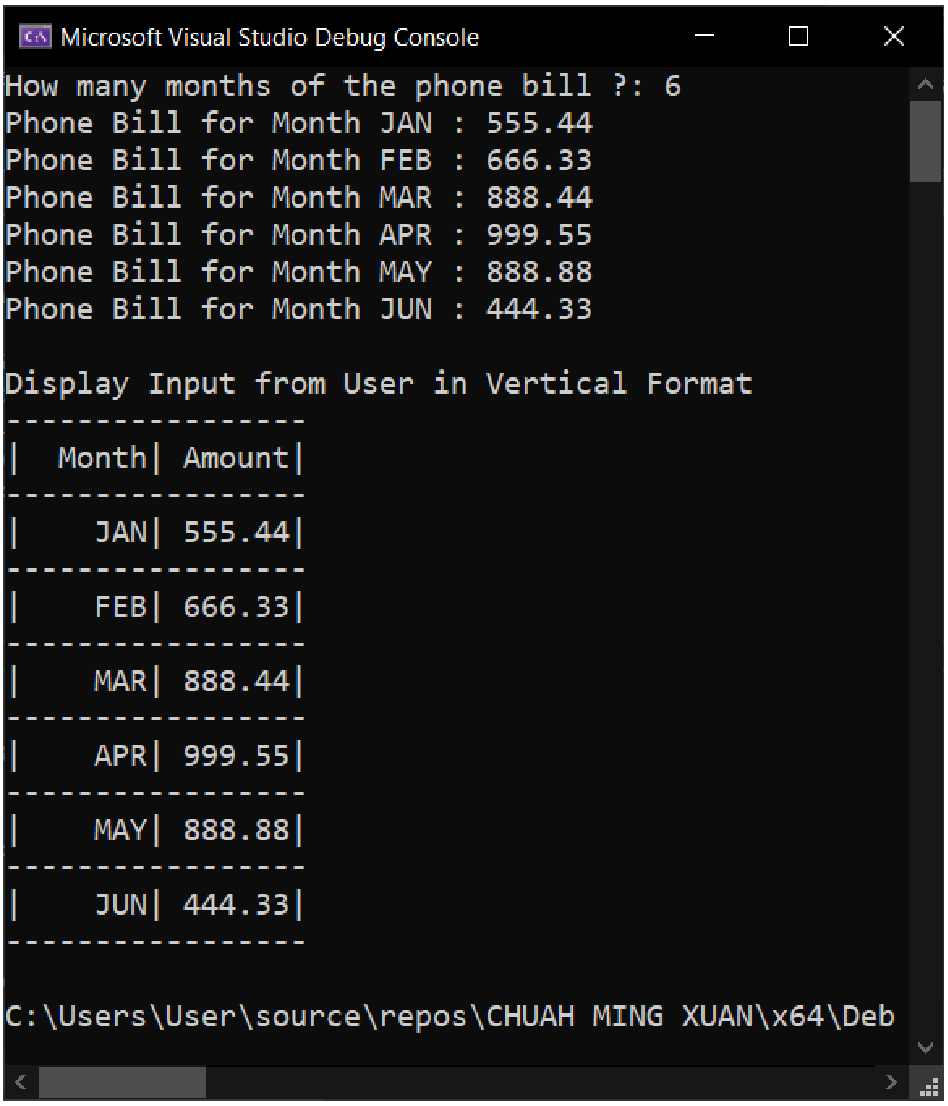
Line Number	Source Code
34	`void vertical(float record[], string row[], int m)`
35	`{ //line 1`
36	`cout << endl << "Display Input from User in Vertical Format" << endl;`
37	`//line 2`
38	`cout << setw(18) << setfill('-') << "\n";`
39	`//line 3 to 16`
40	`for (int i = -1; i < m; i++)`
41	`{`
42	`cout << "|";`
43	`if (i == -1)`
44	`{`
45	`cout << setw(8) << setfill(' ') << "Month|";`
46	`cout << setw(8) << "Amount|";`
47	
48	`}`
49	`else`
50	`{`
51	`cout << setw(7) << setfill(' ') << row[i] << "|";`
52	`cout << setw(7) << record[i] << "|";`
53	`}`
54	`cout << endl;`
55	`cout << setw(18) << setfill('-') << "\n";`
56	`}`
57	`}`
63	
64	`int main()`
65	`{`
66	`float bill[size];`
67	`string monthname[] =`
68	`{ "JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC" };//declare a array with exist values`
69	`int month = input(bill, monthname);`
70	`float sum = calculate(bill, month);`
71	`vertical(bill, monthname, month, sum);`
72	`}`

Table 17. 5 The Relevant Explanation of a C++ Program for Table 17.3 and 17.4

Line Number	Source Code
40	The condition of for loop is changed from `(i<=m)` to `(i<m)`
43-56	The second else `if(i==m)` is removed because the row of total no need to display in the vertical table in this case.
71	The `main` function did not call the `horizontal` function in this case

Iterative 1 - Step 3.2: Test the Program

In this step, let's focus on how to test the program using step-by-step approach. Figure 17.4 shows the screen when the program is executed. The program is waiting for user to input a value.



```
How many months of the phone bill ?: 6
Phone Bill for Month JAN : 555.44
Phone Bill for Month FEB : 666.33
Phone Bill for Month MAR : 888.44
Phone Bill for Month APR : 999.55
Phone Bill for Month MAY : 888.88
Phone Bill for Month JUN : 444.33

Display Input from User in Vertical Format
-----------------
|  Month| Amount|
-----------------
|    JAN| 555.44|
-----------------
|    FEB| 666.33|
-----------------
|    MAR| 888.44|
-----------------
|    APR| 999.55|
-----------------
|    MAY| 888.88|
-----------------
|    JUN| 444.33|
-----------------

C:\Users\User\source\repos\CHUAH MING XUAN\x64\Deb
```

Figure 17. 4 The output of table 17.3 and 17.4

Iterative 2 - Step 3.1: Develop function finding()

As mentioned before ,we have apply basic algorithm to finding the highest and the lowest bill from the array .What we are going to do is we compare the values in the array from the first value to the last value. Then , the comparison ‘s result will pass back to the main function .

Table 17. 6 Develop function finding() -part 1

Line Number	Source Code
1	`//a program that applies 1 d array`
2	`//By Chuah Ming Xuan (D032110227),FTMK ,UTeM`
3	`#include <iostream>`
4	`#include <iomanip>`
5	`using namespace std;`
6	`#define size 12 //the value of size is changed`
7	`//const int size2 = 12;`
8	
9	`int input(float Bill[], string m[])`
10	`{`
11	`        int month; //store the number of the months`
12	`        do {`
13	`                cout << "How many months of the phone bill ?: ";`
14	`                cin >> month;`
15	`                if (month > 0 && month <= size)`
16	`                        break; //exit loop when data in the range`
17	`                cout << "Invalid Input (1-12 only)" << endl;`
18	`        } while (1); //a loop always true is created`
19	`        for (int i = 0; i < month; i++)`
20	`        {`
21	`                cout << "Phone Bill for Month " << m[i] << " : ";`
22	`                cin >> Bill[i];`
23	`        }`
24	`        return month;`
25	`} //input data`
26	
27	`float calculate(float Bill[], int m)`
28	`{`
29	`        float total = 0;`
30	`        for (int i = 0; i < m; total += Bill[i++]);//calculate the total using a loop`
31	`        return total;`
32	`}`
33	

Table 17. 7 Develop function finding() -part 2

Line Number	Source Code
34	void finding(float record[], int m, int& max, int& min)
35	{
36	for (int i = 0; i < m; i++) //check all data
37	{
38	if (i == 0)
39	max = min = i; //set default value for the variables
40	else if (record[i] > record[max])
41	max = i; //max is replaced when another data is higher than it
42	else if (record[i] < record[min])
43	min = i; //min is replaced when another data is lower than it
44	} //in the end of loop ,the highest and lowest value is found
45	}
46	
47	void vertical(float record[], string row[], int m)
48	{ //line 1
49	cout << endl << "Display Input from User in Vertical Format" << endl;
50	//line 2
51	cout << setw(18) << setfill('-') << "\n";
52	//line 3 to 16
53	for (int i = -1; i < m; i++)
54	{
55	cout << "|";
56	if (i == -1)
57	{
58	cout << setw(8) << setfill(' ') << "Month|";
59	cout << setw(8) << "Amount|";
60	
61	}
62	else
63	{
64	cout << setw(7) << setfill(' ') << row[i] << "|";
65	cout << setw(7) << record[i] << "|";
66	}
67	cout << endl;
68	cout << setw(18) << setfill('-') << "\n";
69	}
70	}

Table 17. 8 Develop function finding() -part 3

Line Number	Source Code
71	
72	`int main()`
73	`{`
74	`float bill[size];`
75	`string monthname[] =`
76	`{ "JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC" };//declare a array with exist values`
77	`int month = input(bill, monthname);`
78	`float sum = calculate(bill, month);`
79	`int highest, lowest; //the variable of the highest and lowest bill`
80	`finding(bill, month, highest, lowest);`
81	`vertical(bill, monthname, month, sum);`
82	`}`

Table 17. 9 The Relevant Explanation of a C++ Program for Table 17.6 to 17.8 -part 1

Line Number	Source Code
34	The function `finding` is declared and it will used to find the highest and the lowest bill among the data inside array
36	A for loop is applied to ensure all values are involved in this process
38-39	During the first looping ,the default values of `min` and `max` is assigned as `i` (`i =0` in the first looping)
40-41	If the current values in `record` is higher than the maximum value before ,then the update of the `max` is made
42-43	If the current values in `record` is lower than the minimum value before ,then the update of the `min` is made
79	The variable `highest` and `lowest` is declared
80	The `finding` function is called in the main

Iterative 2 - Step 3.2: No step 3.2 in this iterative

Iterative 3 - Step 3.1: Develop function result()

In the last iterative ,we will develop a function which will display the summary of the phone bills.

Table 17. 10 Develop function result() -part 1

Line Number	Source Code
1	`//a program that applies 1 d array`
2	`//By Chuah Ming Xuan (D032110227),FTMK ,UTeM`
3	`#include <iostream>`
4	`#include <iomanip>`
5	`using namespace std;`
6	`#define size 12 //the value of size is changed`
7	`//const int size2 = 12;`
8	
9	`int input(float Bill[], string m[])`
10	`{`
11	`    int month; //store the number of the months`
12	`    do {`
13	`        cout << "How many months of the phone bill ?: ";`
14	`        cin >> month;`
15	`        if (month > 0 && month <= size)`
16	`            break; //exit loop when data in the range`
17	`        cout << "Invalid Input (1-12 only)" << endl;`
18	`    } while (1); //a loop always true is created`
19	`    for (int i = 0; i < month; i++)`
20	`    {`
21	`        cout << "Phone Bill for Month " << m[i] << " : ";`
22	`        cin >> Bill[i];`
23	`    }`
24	`    return month;`
25	`} //input data`
26	
27	`float calculate(float Bill[], int m)`
28	`{`
29	`    float total = 0;`
30	`    for (int i = 0; i < m; total += Bill[i++]);//calculate the total using a loop`
31	`    return total;`
32	`}`
33	

Table 17. 11 Develop function result() -part 2

Line Number	Source Code
34	void finding(float record[], int m, int& max, int& min)
35	{
36	for (int i = 0; i < m; i++) //check all data
37	{
38	if (i == 0)
39	max = min = i; //set default value for the variables
40	else if (record[i] > record[max])
41	max = i; //max is replaced when another data is higher than it
42	else if (record[i] < record[min])
43	min = i; //min is replaced when another data is lower than it
44	} //in the end of loop ,the highest and lowest value is found
45	}
46	
47	void vertical(float record[], string row[], int m)
48	{ //line 1
49	cout << endl << "Display Input from User in Vertical Format" << endl;
50	//line 2
51	cout << setw(18) << setfill('-') << "\n";
52	//line 3 to 16
53	for (int i = -1; i < m; i++)
54	{
55	cout << "|";
56	if (i == -1)
57	{
58	cout << setw(8) << setfill(' ') << "Month|";
59	cout << setw(8) << "Amount|";
60	
61	}
62	else
63	{
64	cout << setw(7) << setfill(' ') << row[i] << "|";
65	cout << setw(7) << record[i] << "|";
66	}
67	cout << endl;
68	cout << setw(18) << setfill('-') << "\n";
69	}
70	}

Table 17. 12 Develop function result() -part 3

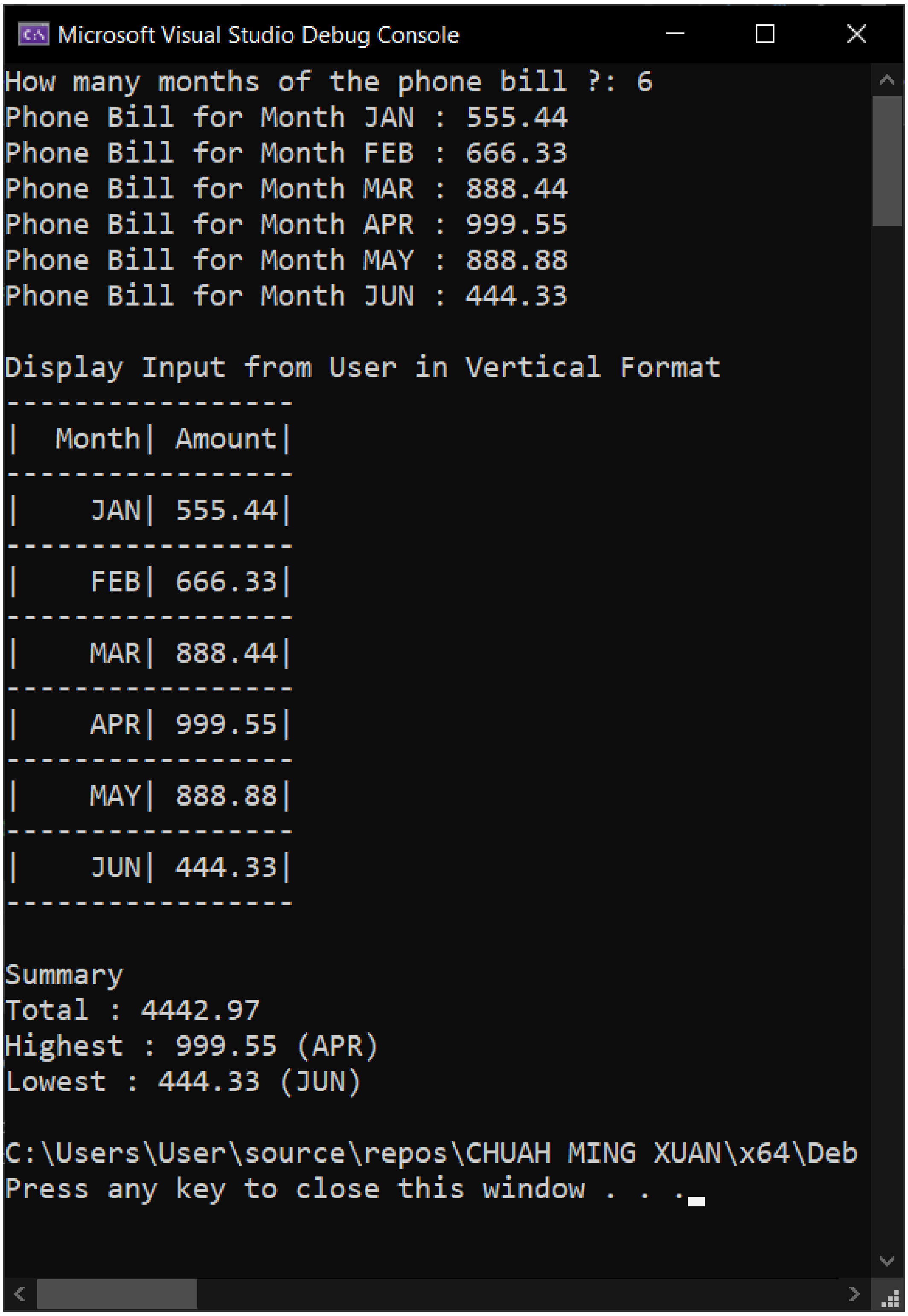
Line Number	Source Code
71	
72	void result(float record[], string name[], float total, int max, int min)
73	{
74	cout << endl << "Summary" << endl;
75	cout << "Total : "<< total << endl;
76	cout << "Highest : " << record[max] << " (" << name[max] << ")" << endl;
77	Cout << "Lowest : " << record[min] << " (" << name[min] << ")" << endl;
78	} //display the result
79	
79	int main()
80	{
81	float bill[size];
82	string monthname[] =
83	{ "JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC" };//declare a array with exist values
84	int month = input(bill, monthname);
85	float sum = calculate(bill, month);
86	int highest, lowest; //the variable of the highest and lowest bill
87	finding(bill, month, highest, lowest);
88	vertical(bill, monthname, month, sum);
89	result(bill, monthname, sum, highest, lowest);
90	}

Table 17. 13 The Relevant Explanation of a C++ Program for Table 17.3 and 17.4

Line Number	Source Code
72-78	This function is developed to display the summary
89	The main function call the result function

Iterative 3 - Step 3.2: Test the Program

In this step, let's focus on how to test the program using step-by-step approach. Figure 17.4 shows the screen when the program is executed. The program is waiting for user to input a value.



```
How many months of the phone bill ?: 6
Phone Bill for Month JAN : 555.44
Phone Bill for Month FEB : 666.33
Phone Bill for Month MAR : 888.44
Phone Bill for Month APR : 999.55
Phone Bill for Month MAY : 888.88
Phone Bill for Month JUN : 444.33

Display Input from User in Vertical Format
-----------------
|  Month| Amount|
-----------------
|    JAN| 555.44|
-----------------
|    FEB| 666.33|
-----------------
|    MAR| 888.44|
-----------------
|    APR| 999.55|
-----------------
|    MAY| 888.88|
-----------------
|    JUN| 444.33|
-----------------

Summary
Total : 4442.97
Highest : 999.55 (APR)
Lowest : 444.33 (JUN)

C:\Users\User\source\repos\CHUAH MING XUAN\x64\Deb
Press any key to close this window . . .
```

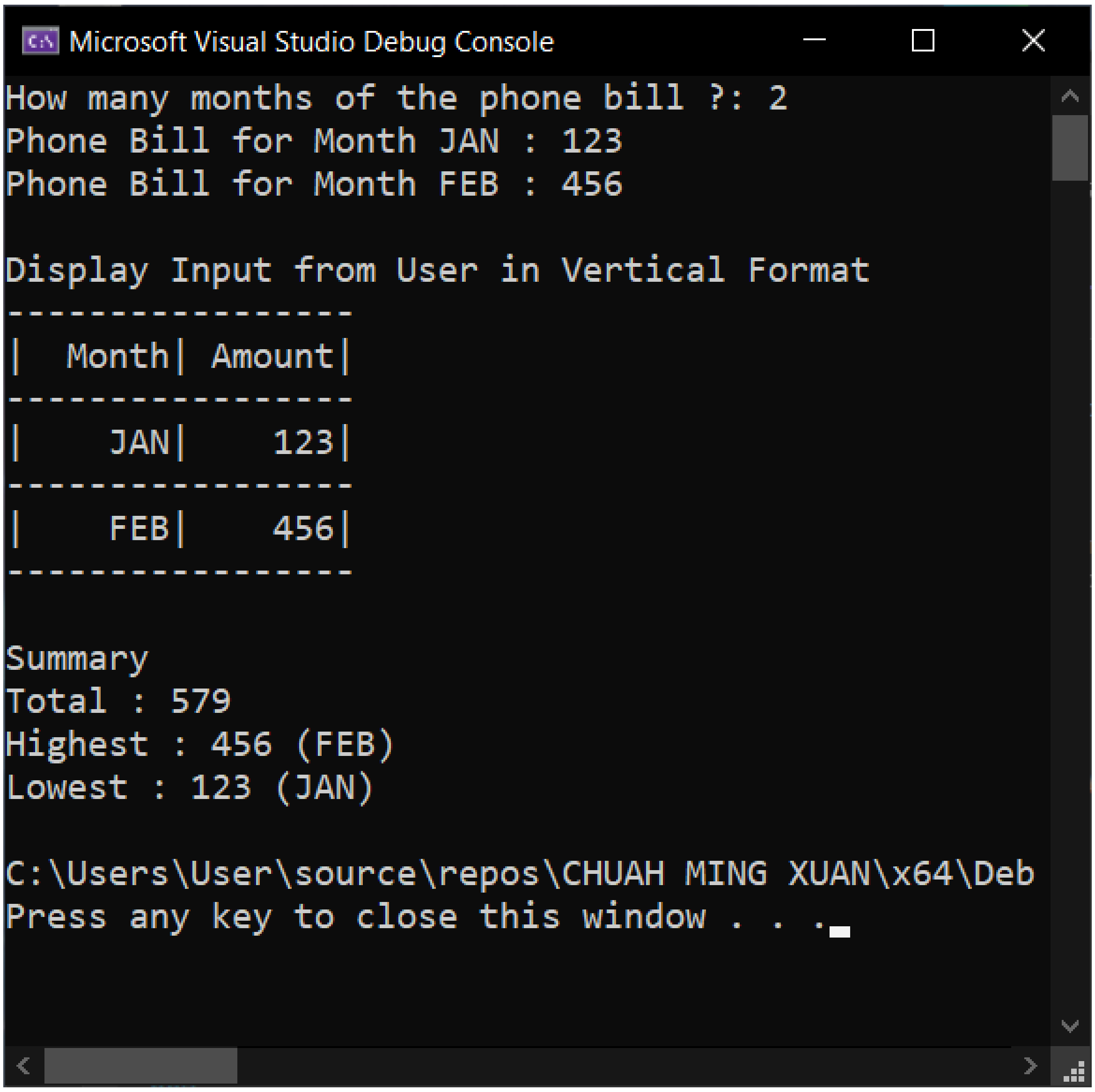
Figure 17. 5 the output of table 17.10 to 17.12

Step 4: Verify the Program

This step is to compare the expected output from the question and the output generated as result of the execution of the program. Output must satisfy between our expectation as shown in Table 17.14 and outputs shown in Figure 15.10 and Figure 15.11.

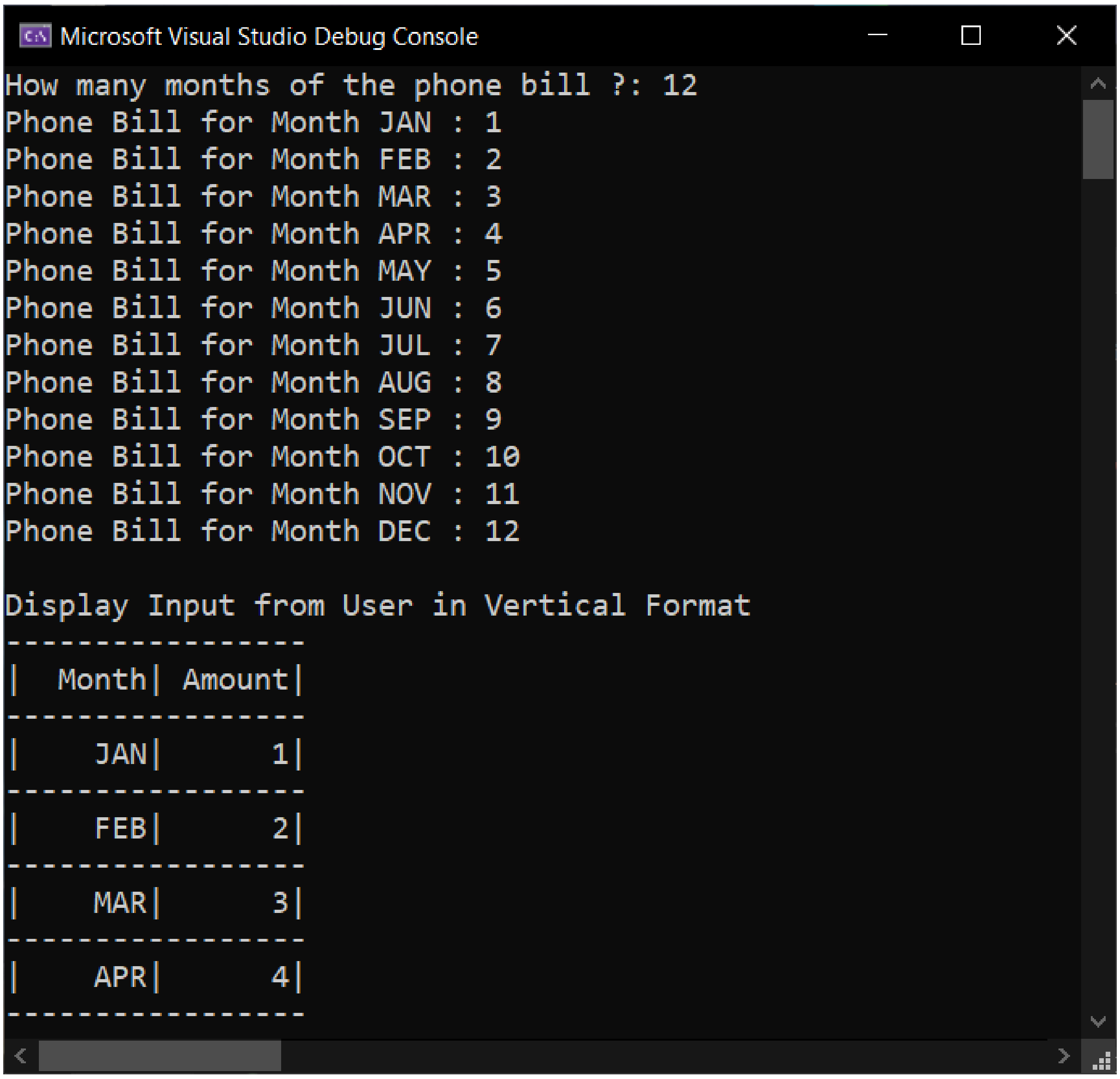
Table 17. 14 The Example of Verification of the Formula Based on New Test Case

	Output 1	**New Test Case 1**	**New Test Case 2**
Number of month	6	2	12
Bill	bill[0] 555.44 bill[1] 666.33 bill[2] 888.44 bill[3] 999.55 bill[4] 888.88 bill[5] 444.33	bill[0] 123 bill[1] 456	bill[0] 1 bill[1] 2 bill[2] 3 bill[3] 4 bill[4] 5 bill[5] 6 bill[6] 7 bill[7] 8 bill[8] 9 bill[9] 10 bill[10] 11 bill[11] 12
Total	4442.97	579.00	78.00
Highest	999.55	456	12
Lowest	444.33	123	1


Microsoft Visual Studio Debug Console

```
How many months of the phone bill ?: 2
Phone Bill for Month JAN : 123
Phone Bill for Month FEB : 456

Display Input from User in Vertical Format
-----------------
|  Month| Amount|
-----------------
|    JAN|    123|
-----------------
|    FEB|    456|
-----------------

Summary
Total : 579
Highest : 456 (FEB)
Lowest : 123 (JAN)

C:\Users\User\source\repos\CHUAH MING XUAN\x64\Deb
Press any key to close this window . . .
```

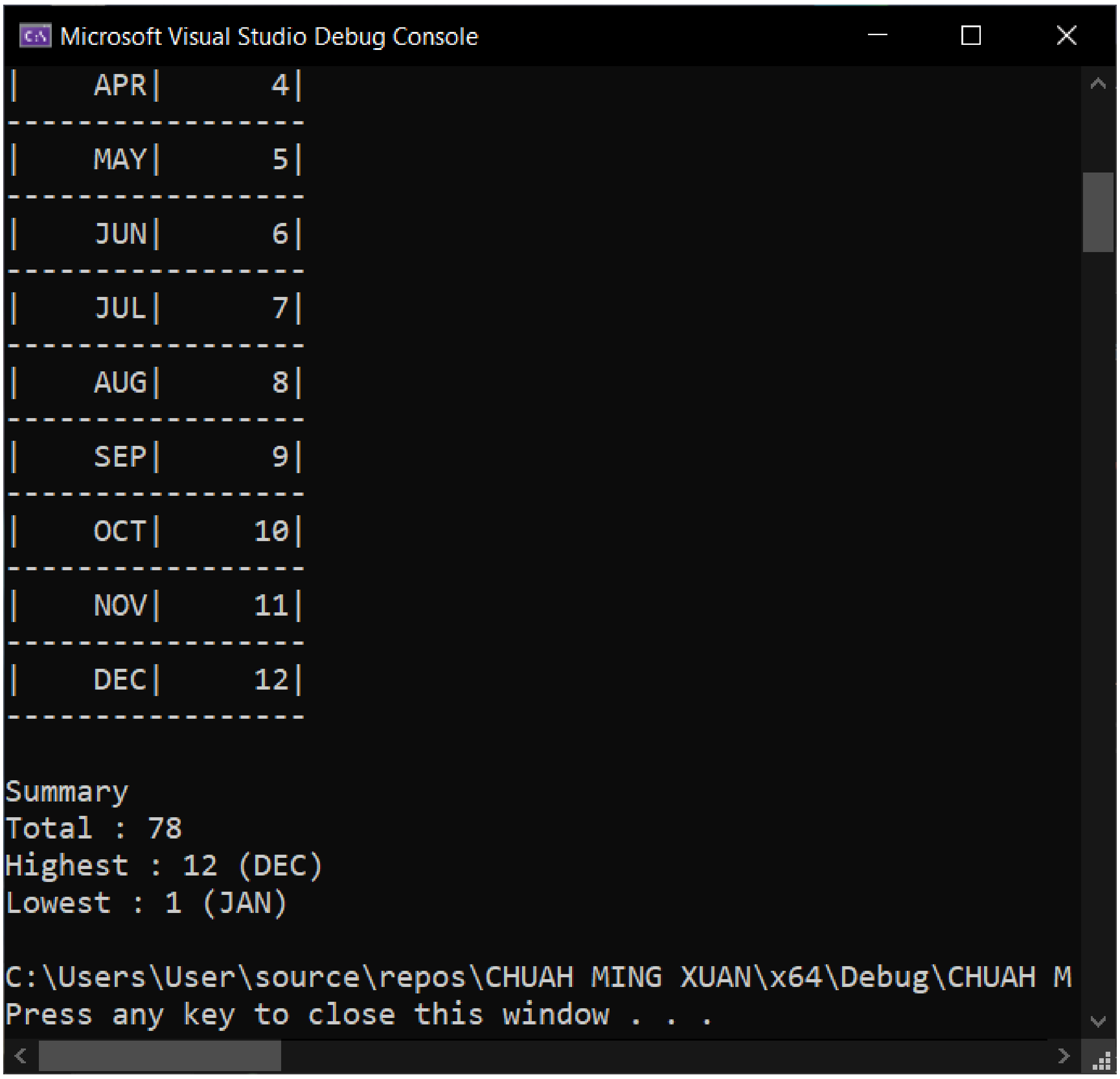
Figure 17. 6 Output from Execution of the New Test Case 1



Microsoft Visual Studio Debug Console

```
How many months of the phone bill ?: 12
Phone Bill for Month JAN : 1
Phone Bill for Month FEB : 2
Phone Bill for Month MAR : 3
Phone Bill for Month APR : 4
Phone Bill for Month MAY : 5
Phone Bill for Month JUN : 6
Phone Bill for Month JUL : 7
Phone Bill for Month AUG : 8
Phone Bill for Month SEP : 9
Phone Bill for Month OCT : 10
Phone Bill for Month NOV : 11
Phone Bill for Month DEC : 12

Display Input from User in Vertical Format
-----------------
|  Month| Amount|
-----------------
|    JAN|      1|
-----------------
|    FEB|      2|
-----------------
|    MAR|      3|
-----------------
|    APR|      4|
-----------------
```

Figure 17. 7 Output from Execution of the New Test Case 2 -part 1



```
|    APR|       4|
-----------------
|    MAY|       5|
-----------------
|    JUN|       6|
-----------------
|    JUL|       7|
-----------------
|    AUG|       8|
-----------------
|    SEP|       9|
-----------------
|    OCT|      10|
-----------------
|    NOV|      11|
-----------------
|    DEC|      12|
-----------------

Summary
Total : 78
Highest : 12 (DEC)
Lowest : 1 (JAN)

C:\Users\User\source\repos\CHUAH MING XUAN\x64\Debug\CHUAH M
Press any key to close this window . . .
```

Figure 17. 8 Output from Execution of the New Test Case 2 -part 2

17.2 Explanation of Problem Solving Steps

Step 1: Understand the Problem
Here ,we are still using the same tools to understand the problem.

Step 2: Plan and Design Solution
For chapter 15 onwards, we assumed that students are familiar how to plan and design a solution. They are able to visualise CIPO chart while implementing a solution. We move on to Step 3.

Step 3: Implement the Solution
In this step ,we will discuss the process completed in `finding` function which applies a basic algorithmn.

Table 17. 15 Source code of function `finding`

Line Number	**Source Code**
1	`void finding(float record[], int m, int& max, int& min)`
2	`{`
3	`    for (int i = 0; i < m; i++) //check all data`
4	`    {`
5	`        if (i == 0)`
6	`            max = min = i; //set default value for the variables`
7	`        else if (record[i] > record[max])`
8	`            max = i; //max is replaced when another data is higher than it`
9	`        else if (record[i] < record[min])`
10	`            min = i; //min is replaced when another data is lower than it`
11	`    } //in the end of loop ,the highest and lowest value is found`
12	`}`

Table 17. 16 Explanation of Source Code of Function `finding` - part 1

Line Number	Source Code
5-6	The default values of `max` and `min` is assigned to the first value in the array due to no comparison is made between values.
7-8	If the current value in the variable `max` is lower than the value in the array which is fetched from the subsequent loop ,then the value of `max` is updated so that `max` is always store the index number which has the highest values among the previous values.
9-10	The same method is used to find the lowest bill . If the current value in the variable `max` is higher than the value in the array which is fetched from the subsequent loop ,then the value of `min` is updated so that `min` is always store the index number which has the lowest values among the previous values
1-12	Call the function `finding()` in main There is some explanation pertaining to the algorithm we used in the function finding(). Assumed that these are the values entered by the user MONTH: BILL JAN: 50 FEB: 100 MAR: 70 APR: 10 The update of variable max and min in the function finding() :

MONTH	BILL
JAN	50
FEB	100
MAR	70
APR	10

Loop (value of i)	0	1	2	3
The value fetched (record [i])	50	100	70	10
max	0	1	1	1
record[max]	50	100	100	100
min	0	0	0	3
record[min]	50	50	50	10

Table 17. 17 Explanation of Source Code of Function `finding` - part 2

Line Number	Source Code
1-12	When `i` is equal to 0 (first loop) ,the value of *max* and *min* is assigned as 0 (which is current value of `i`) When `i` become 1 due to `i++` executed in the end of the loop ,the value fetched `(record[i])` is 100 .The condition `else if (record[i] > record[max])` is satisfied .Hence ,the value of `max` is overwrited to 1 . When `i` increase to 2 , the value fetched `(record[i])` is 70 . Neither the condition `else if (record[i] > record[max])` nor `else if (record[i] < record[min])` is satisfied .Hence ,no manipulation of value happened in this looping . When `i` is altered to 3 , the value fetched `(record[i])` is 10 .The condition `else if (record[i] < record[min])` is satisfied .Hence ,the value of `min` is changed to 3 . As the result ,the highest bill and lowest bill is obtained from this function via this comparison algorithm from the first value to the last value. Here comes another question ,why doesn’t we just simply store the value of the highest and lowest bill ? We can noticed that the variables `max` and `min` is used to store the index number of the array but not the values inside array . `//assume that max and min is float datatype now` `if (i == 0)` `    max = min = record[i];` `else if (record[i] > record[max])` `    max = record[i];` `else if (record[i] < record[min])` `    min = record[i];` The codes above will store the values inside array into variable `max` and `min` .It is simply to implement and looks more make sense than we just assigned the index number into these both variables. However ,we will get into troubles when we want to display the month name which has highest bill and lowest bill .The only approach to resolve this problem is we have to declared some new variables again to save the month name of the highest bill and lowest bill .Thus ,you are advocated to store the index number into `max` and `min` so that we can apply them into other array easily.

Step 4: Verify the Program
It is also advisable to test other values in the our programme so that the error can be noticed and eliminated.

17.3 Summary

The learning outcomes that we achieve in this chapter is we can practice the algorithm inside our program .Finding and searching the highest value and lowest value from a collection of data (array) is one of most simplest algorithm .There are a lot of algorithm that we will continue to learn and to handle in the future so that our program can tackle more and more complicated situations .However ,once we understand the concept of if-else statement ,looping and array well ,you will not take too long time to clarify the confusion arisen when you come across some strange algorithm.

17.4 Question

In this chapter there some questions are to test problem solving in programming

17.4.1 Problem Solving in Programming

Students are required to follow the steps that have been discussed. Your answer must be in the form of steps and iteration. You also need **to write relevant and useful comments** in your program.

Question 1

Using the programme that has been develop in this chapter , find out the months which are the second highest amount and second lowest amount .The number of month in the programme is limited at least 5 and at most 12.

Question 2

Write a C++ program to record 10 students' mark in a programming lab test. Then ,your programme should be able to display all excellent students and their mark in vertical format in the end of the programme. The excellent students are the students who get the mark 20% higher than the average mark of all students.

Example:

Student	0	1	2	3	4	5	6	7	8	9
Mark	0	10	20	30	40	50	60	70	80	90
Total	0 + 10 + 20 + 30 + 40 + 50 + 60 + 70 + 80 + 90 = 450									
Average	450 /10 = 45									
Excellent student mark	45 x 1.2 =54									

From the result of the calculation student 6 ,7,8 ,9 are the excellent students because their marks are higher than 54

Question 3

Write a C++ program which enables user to enter 10 integer into an array. Then, the positions of the array are swapped each other .The first element and the last element will swapped position ,the second element and the second last element swapped position .Then ,your program should be able to display table below

Before Swapping	1	2	3	4	5	0	0	0	0	0
After Swapping	0	0	0	0	0	5	4	3	2	1

*Hint : The row of before swapping and the row after swapping are mirrorlike.