

## Activity No. 1.1

### Hands-on Activity 2.1: Data Types and Arithmetic Operations

**Course Code:** CPE010

**Program:** Computer Engineering

**Course Title:** Data Structures and Algorithms

**Date Performed:**

**Section:** CPE11S1

**Date Submitted:**

**Name(s):** Tobias, Lawrence C.

**Instructor:** Engr. Jimlord M. Quejado

#### 6. Output

**Example 1:** The following program has an output of:

The value of seven is: 7.000000

The value of eight and a half is: 8.500000

Can you find all possible compilation errors and logic errors? Can you fix them to print the same result as the expected output? Before you use your compiler, try to find the errors only by manual code analysis.

- The original code had errors such as missing output for the number 7, incorrect use of a comma instead of the <<operator, and no formatting for decimal places. By declaring float variables for the values, using `#include <iomanip>`, and applying `fixed` and `setprecision(6)`, the corrected program properly displays the expected output with six decimal places.

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
cout<<"The value of seven is: ";
```

```
cout<<"The value of eight and a half is: ", <<8.5;
```

```
return 0;
```

```
}
```

main.cpp	Output
<pre> 1  #include&lt;iostream&gt; 2  #include&lt;iomanip&gt; 3  int main() 4  { 5 6  float cpe11s1 = 7.000000; 7  float cpe11s2 = 8.500000; 8 9  std::cout&lt;&lt;"The value of seven is:" &lt;&lt;std::fixed&lt;&lt;std::setprecision(6)&lt;&lt;cpe11s1&lt;&lt;std::endl 10 ; 11 std::cout&lt;&lt;"The value of eight and a half is: " &lt;&lt;cpe11s2&lt;&lt;std::endl; 12 return 0; 13 14 }</pre>	<pre> The value of seven is:7.000000 The value of eight and a half is: 8.500000  === Code Execution Successful ===</pre>

**Example 2:** The following program has an output of:

The value of seven is: 7.000000

The value of eight and a half is: 8.500000

Can you find all possible compilation errors and logic errors? Can you fix them to print the same result as the expected output? Before you use your compiler, try to find the errors only by manual code analysis.

- The original code had errors such as the 70 has a space and it has no decimal points.

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
cout<<"The value of seven is: "<< 7 0;
```

```
cout<<"The value of eight and a half is: "<<8.5;
```

```
return 0;
```

```
}
```

main.cpp	Output
<pre> 1  #include&lt;iostream&gt; 2  #include&lt;iomanip&gt; 3  int main() 4  { 5 6  float cpe11s1 = 7.000000; 7  float cpe11s2 = 8.500000; 8 9  std::cout&lt;&lt;"The value of seven is:" &lt;&lt;std::fixed&lt;&lt;std::setprecision(6)&lt;&lt;cpe11s1&lt;&lt;std::endl 10 ; 11 std::cout&lt;&lt;"The value of eight and a half is: " &lt;&lt;cpe11s2&lt;&lt;std::endl; 12 return 0; 13 14 }</pre>	<pre> The value of seven is:7.000000 The value of eight and a half is: 8.500000  === Code Execution Successful ===</pre>

**Example 3:** The following program has an output of:

The value of half is: 0.500000

The value of Pi is: 3.141593

Can you find all possible compilation errors and logic errors? Can you fix them to print the same result as the expected output? Before you use your compiler, try to find the errors only by manual code analysis.

- I fixed the code by correcting the variable names, removing spaces in the float values, adding `#include <iomanip>`, and using `fixed` with `setprecision(6)` to format the output. Now, the program correctly displays the values of half and Pi with six decimal places.

```

int main()
{
float halfValue = 0.6;
float piValue = 3.141 592 65;
cout<<"The value of half is: "<< half Value;
cout<<"The value of Pi is: "<<pi_Value;
return 0;
}
```

<div>main.cpp</div> <pre>1 #include&lt;iostream&gt; 2 #include&lt;iomanip&gt; 3 int main() 4 5 { 6 7     float halfValue = 0.500000; 8     float piValue = 3.14159265; 9 10    std::cout&lt;&lt;"The value of half is: "&lt;&lt; std::fixed&lt;&lt;std::setprecision(6)&lt;&lt;halfValue&lt;&lt;std         ::endl; 11 12    std::cout&lt;&lt;"The value of Pi is: "&lt;&lt; piValue; 13 14    return 0; 15 16 }</pre>	<div>Output</div> <pre>The value of half is: 0.500000 The value of Pi is: 3.141593  === Code Execution Successful ===</pre>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------

#### Example 4: Sample program for Adding Two Integers

- This program asks to input two integers, then adds them together and displays the result. It uses cin to take input and cout to print the sum with a clear message.

```
#include <iostream>

int main()
{
    int integer1, integer2, sum; /*declaration */
    cout<<"Enter first integer: \n"; /* prompt */
    cin>>integer1 ;    /* read an integer */
    cout<<"Enter second integer: \n"; /* prompt */
    cin<<integer2;    /* read an integer */
    sum = integer1 + integer2;    /* assignment of sum */
    cout<<"Sum is : "<<sum;    /* print sum */

    return 0; /* indicate that program ended successfully */
}
```

main.cpp

Share

Run

```

1  #include <iostream>
2
3  int main()
4  {
5      int integer1;
6      int integer2;
7      int sum = integer1 + integer2;
8      std::cout<<"Enter first integer:\n";
9      std::cin>>integer1;
10     std::cout<<"Enter second integer:\n";
11     std::cin>>integer2;
12     sum = integer1 + integer2;
13     std::cout << "Sum is " << sum <<std::endl;
14     return 0;
15 }
16

```

Enter first integer:

13

Enter second integer:

13

Sum is 26

=== Code Execution Successful ===

## 7. Supplementary Activity

1. Take a look at the code below: it assigns two integer values, manipulates them and finally outputs the result and bigresult variables. The problem is that the manipulations have been described using natural language, so the code is completely useless now. Act as an intelligent (naturally!) compiler and translate the formula into a real "C" code notation. Test your code using the data provided.

```
#include <iostream>
```

```
using namespace std;
```

```
int main(void)
```

```
{
```

```
int xValue=5;
```

```
int yValue=9;
```

```
int result;
```

```
int bigResult;
```

```
/*
```

```
increment xValue by 3
```

```
decrement yValue by xValue
```

```
multiply xValue times yValue giving result
```

```
increment result by result
```

```
decrement result by 1
```

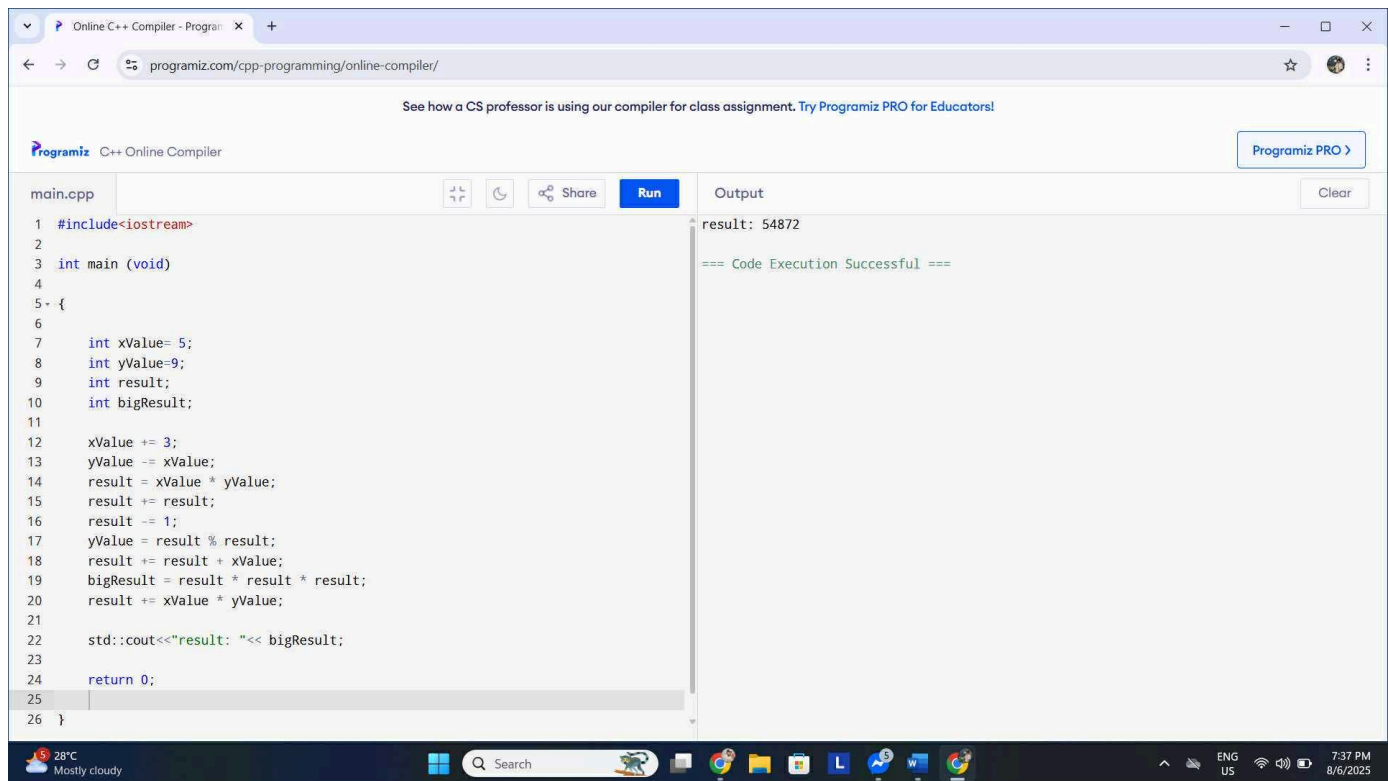
```
assign result modulo result to yValue
```

```
increment result by result added to xValue
```

```
assign result times result times result to bigResult increment result by xValue times yValue
```

\*/

```
cout<<"result: "<<result;
cout<<"big result: "<< bigResult;
return 0;
}
```



The screenshot shows a web browser window with the URL `programiz.com/cpp-programming/online-compiler/`. The page title is "Online C++ Compiler - Program". The main content area displays a C++ program in a text editor, with line numbers 1 through 26. The program calculates a result and a big result. The output window on the right shows the result: 54872, followed by "=== Code Execution Successful ===". The bottom of the browser window shows a Windows taskbar with the date 8/6/2025 and time 7:37 PM.

```
main.cpp
1 #include<iostream>
2
3 int main (void)
4 {
5 {
6
7     int xValue= 5;
8     int yValue=9;
9     int result;
10    int bigResult;
11
12    xValue += 3;
13    yValue -= xValue;
14    result = xValue * yValue;
15    result += result;
16    result -= 1;
17    yValue = result % result;
18    result += result + xValue;
19    bigResult = result * result * result;
20    result += xValue * yValue;
21
22    std::cout<<"result: "<< bigResult;
23
24    return 0;
25 }
26 }
```

Output

```
result: 54872
=== Code Execution Successful ===
```

2. Complete the program below. Compute the accrued amount of money with a starting value of 100 and an annual interest rate of 1.5%. Compute and print the results for first three years. Your version of the program must print the same result as the expected output for every year. Compute each annual value on the basis of the previous year's value.

```
#include <iostream>
using namespace std;
int main()
{
float startValue = 100;
float interestRate = 0.015;
float firstYearValue;
```

```

float secondYearValue;

float thirdYearValue;

/* Your code */

cout<<"After first year: "<<firstYearValue;
cout<<"After second year: "<<secondYearValue; cout<<"After third year: "<<thirdYearValue;
return 0;
}

```

### Example output

After first year: 101.500000

After second year: 103.022499

After third year: 104.567833

The screenshot shows the Programiz C++ Online Compiler interface. The code editor on the left contains the following C++ code:

```

1 #include<iostream>
2
3 int main()
4 {
5
6     float startValue = 100;
7     float interestRate = 0.015;
8
9     float firstYearValue;
10    float secondYearValue;
11    float thirdYearValue;
12
13    firstYearValue = startValue * (1 + interestRate);
14    secondYearValue = firstYearValue * (1 + interestRate);
15    thirdYearValue = secondYearValue * (1 + interestRate);
16
17    std::cout<<"After first year: "<<firstYearValue << std::endl;
18    std::cout<<"After second year: "<<secondYearValue << std::endl;
19    std::cout<<"After third year: "<<thirdYearValue << std::endl;
20 }

```

The output window on the right displays the results of the code execution:

```

After first year: 101.5
After second year: 103.022
After third year: 104.568

```

Below the output, it says "=== Code Execution Successful ===". The browser's address bar shows the URL "programiz.com/cpp-programming/online-compiler/". The bottom of the image shows a Windows taskbar with the date and time "7:56 PM 8/6/2025".

## 8. Conclusion

I now know and have learned that when you combine symbols, characters, operators, and other elements, you can create something interesting like math solutions in C++. Even when working with percentages, you can still solve problems easily as long as you understand how to use these parts properly. What's important is knowing and learning how each one works and where to place them in your code. It may look hard at first, but once you get the hang of it, solving math problems using code becomes fun and exciting.

## 9. Assessment Rubric

