Activity No. 2.1	
Data Types and Arithmetic Operations	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 8/6**
Section: CPE11S1	**Date Submitted: 8/6**
Name(s): Juan Paulo C. Lara	**Instructor: Engr. Jimlord M. Quejado**

6. Output

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.
   The Result should be:
   result: 38
   big result: 54872

   Code:

main.cpp

```cpp
#include <iostream>
using namespace std;

int main(void)

{
    int xValue = 5;
    int yValue = 9;

    int result;
    int bigResult;

    xValue += 3;
    yValue -= xValue;

    result = xValue * yValue;
    result += result;
    result--;

    yValue = result % result;
    result += (result + xValue);

    bigResult = result * result * result;
    result += xValue * yValue;

    cout << "result: " << result << endl;
    cout << "big result: " << bigResult << endl;

    return 0;
}
```

Console

```
result: 38
big result: 54872
```

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.
   Expected output:
   After first year: 101.500000
   After second year: 103.022499
   After third year: 104.567833

main.cpp

```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    float startValue = 100.0;
    float interestRate = 0.015;

    float firstYearValue;
    float secondYearValue;
    float thirdYearValue;

    firstYearValue = startValue + startValue * interestRate;
    secondYearValue = firstYearValue + firstYearValue * interestRate;
    thirdYearValue = secondYearValue + secondYearValue * interestRate;

    cout << fixed << setprecision(6);
    cout << "After first year: " << firstYearValue << endl;
    cout << "After second year: " << secondYearValue << endl;
    cout << "After third year: " << thirdYearValue << endl;

    return 0;
}
```

Console

```
After first year: 101.500000
After second year: 103.022499
After third year: 104.567833
```

7. Supplementary Activity

1.

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.
Sample:

main.cpp

```cpp
#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;

}
```

Console

```
main.cpp: In function ‘int main()’:
main.cpp:11:45: error: expected primary-expression before ‘<<’ token
 cout<<"The value of eight and a half is: ", <<8.5;
```

The error in this code is within line 11. The “;” or statement terminator is placed before the “<<” left shift operator. The absence of “endl” can lead to two statements occurring at the same line. Even if the errors were fixed, the values do not match the given output above. This requires including the iomanip library and setting a precision amount to achieve the required output.

Fixed:

main.cpp

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    cout << fixed << setprecision(6);

    cout << "The value of seven is: " << 7.0 << endl;

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

    return 0;
}
```

Console

```
The value of seven is: 7.000000
The value of eight and a half is: 8.500000
```

2.

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 codes between 1 and 2 are very similar but there are some noticeable differences. Line 11 no longer has the terminating statement before 8.5. But both share the lack of “endl” statements. The decimals may not be considered an error but it is required to match the expected output.

Sample:

main.cpp

```
#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;

}
```

Console

```
main.cpp: In function ‘int main()’:
main.cpp:9:37: error: expected ‘;’ before numeric constant
 cout<<"The value of seven is: "<< 7 0;
```

Fixed:

main.cpp

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    cout << fixed << setprecision(6);

    cout << "The value of seven is: " << 7.0 << endl;

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

    return 0;
}
```

Console

```
The value of seven is: 7.000000
The value of eight and a half is: 8.500000
```

3.

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.

The most obvious error is within line 7 containing spaces between the numbers which can cause problems when compiling. Even though we can recognize the decimal places, the computer or compiler in this case, does not. Also, the lack of “endl” statements also cause a visual mismatch between this code’s output and the given output.

Sample:

main.cpp

```
1  int main()
2
3  {
4
5  float halfValue = 0.6;
6
7  float piValue = 3.141 592 65;
8
9  cout<<"The value of half is: "<< half Value;
10
11 cout<<"The value of Pi is: "<<pi_Value;
12
13 return 0;
14
15 }
```

Console

```
main.cpp: In function ‘int main()’:
main.cpp:7:23: error: expected ‘,’ or ‘;’ before numeric constant
 float piValue = 3.141 592 65;

main.cpp:9:1: error: ‘cout’ was not declared in this scope
 cout<<"The value of half is: "<< half Value;

main.cpp:9:34: error: ‘half’ was not declared in this scope
 cout<<"The value of half is: "<< half Value;

main.cpp:11:31: error: ‘pi_Value’ was not declared in this scope
 cout<<"The value of Pi is: "<<pi_Value;
```

Fixed:

main.cpp

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4
5  int main()
6  {
7      float halfValue = 0.5;
8      float piValue = 3.141593;
9
10     cout << fixed << setprecision(6);
11     cout << "The value of half is: " << halfValue << endl;
12     cout << "The value of Pi is: " << piValue << endl;
13
14     return 0;
15 }
```

Console

```
The value of half is: 0.500000
The value of Pi is: 3.141593
```

4. Example 4: Sample program for Adding Two Integers
Sample Output:
Enter first integer
45
Enter second integer
72
Sum is 117

As you can see in the console, there are stray errors. This is the result of the quotation marks being changed to left and right facing quotations, which the compiler does not understand. This is more of an error you’d encounter when copying code from processed texts in webpages. The absence of “using namespace std;” also leads to undeclared statements in the scope.

Sample:

main.cpp

```
1  #include <iostream>
2
3  int main()
4
5  {
6
7  int integer1, integer2, sum; /*declaration */
8
9  cout<<“Enter first integer: \n” ;  /* prompt */
10
11 cin>>integer1 ;           /* read an integer */
12
13 cout<<“Enter second integer: \n” ; /* prompt */
14
15 cin<<integer2;           /* read an integer */
16
17 sum = integer1 + integer2;          /* assignment of sum */
18
19 cout<<“Sum is : "<<sum;        /* print sum */
20
21
22
23 return 0;   /* indicate that program ended successfully */
24
25 }
```

Console

```
main.cpp:9:1: error: stray ‘\342’ in program
 cout<<“Enter first integer: \n” ;  /* prompt */

main.cpp:9:1: error: stray ‘\200’ in program
main.cpp:9:1: error: stray ‘\234’ in program
main.cpp:9:1: error: stray ‘\’ in program
main.cpp:9:1: error: stray ‘\342’ in program
main.cpp:9:1: error: stray ‘\200’ in program
main.cpp:9:1: error: stray ‘\235’ in program
main.cpp:13:1: error: stray ‘\342’ in program
 cout<<“Enter second integer: \n” ; /* prompt */

main.cpp:13:1: error: stray ‘\200’ in program
main.cpp:13:1: error: stray ‘\234’ in program
main.cpp:13:1: error: stray ‘\’ in program
main.cpp:13:1: error: stray ‘\342’ in program
main.cpp:13:1: error: stray ‘\200’ in program
main.cpp:13:1: error: stray ‘\235’ in program
main.cpp:19:1: error: stray ‘\342’ in program
 cout<<“Sum is : "<<sum;        /* print sum */
```

Fixed:

main.cpp

```
#include <iostream>
using namespace std;
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 << endl;        /* print sum */

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

}
```

Console

```
Enter first integer
45
Enter second integer
72
Sum is 117
```

8. Conclusion

Concluding this activity, I have learned in finding possible errors in logic and compilation, and completing programs by applying the appropriate data types and arithmetic operations. With this, I am able to make corrections in code and logic errors to make them execute an output correctly. The errors in the console also taught me how to find the errors and find ways to make them work. There were also specific requirements within the expected output like a certain amount of decimals of an outputted value. To match the given output, I researched other C++ libraries such as iomanip, which helped me match the output’s values to the given. Overall, this activity has allowed me to further expand my knowledge in C++ as a beginner.

9. Assessment Rubric