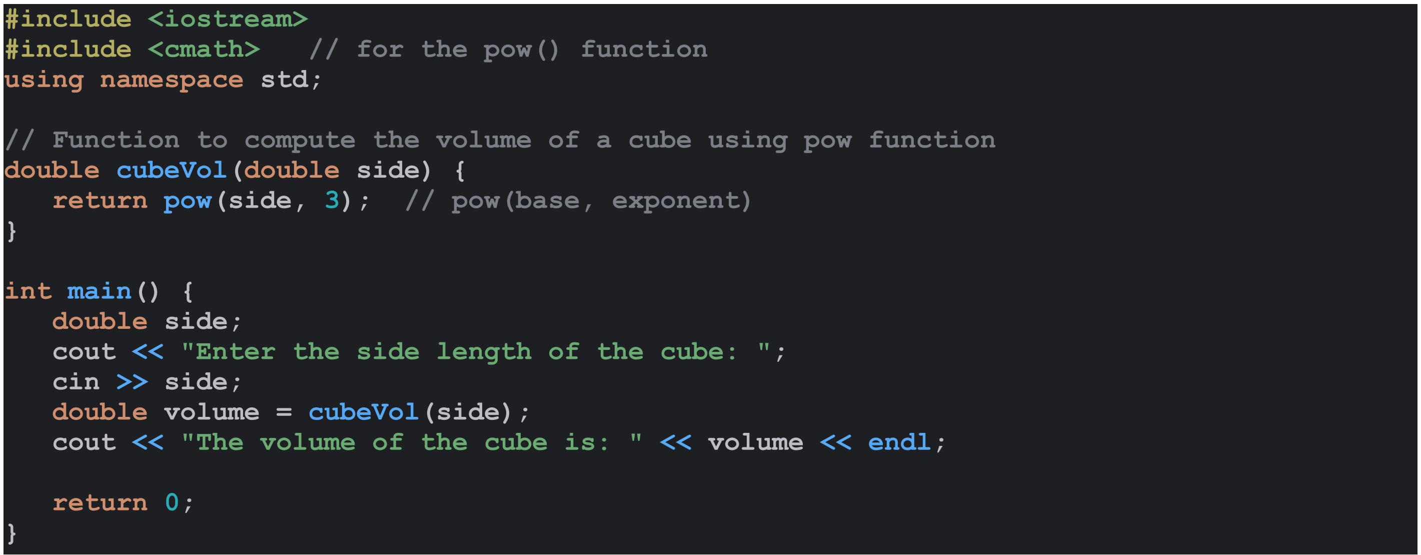
Activity 6.2	
Built-in Functions	
Course Code: CPE 007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: October 23, 2025**
Section: CPE11S1	**Date Submitted: October 28, 2025**
Name(s): Jaime Luis M. Demain	**Instructor: Engr. Jimlord M. Quejado**
6. Output	

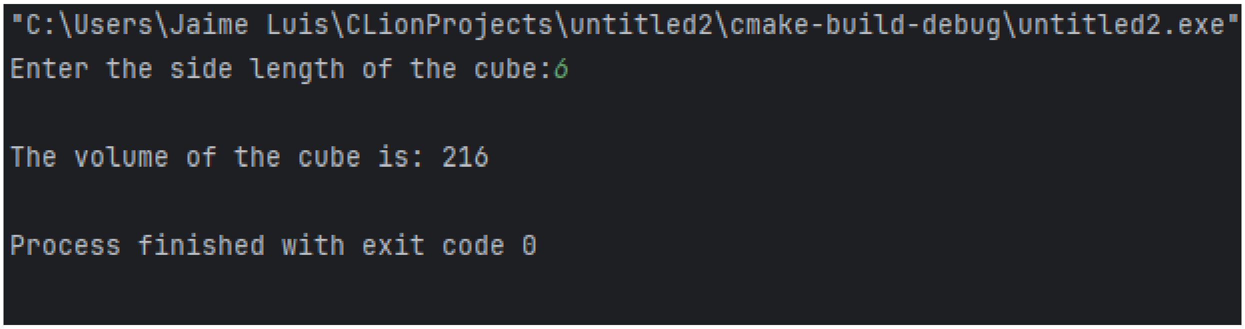
1.

Code:



```cpp
#include <iostream>
#include <cmath>    // for the pow() function
using namespace std;

// Function to compute the volume of a cube using pow function
double cubeVol(double side) {
   return pow(side, 3);  // pow(base, exponent)
}

int main() {
   double side;
   cout << "Enter the side length of the cube: ";
   cin >> side;
   double volume = cubeVol(side);
   cout << "The volume of the cube is: " << volume << endl;

   return 0;
}
```

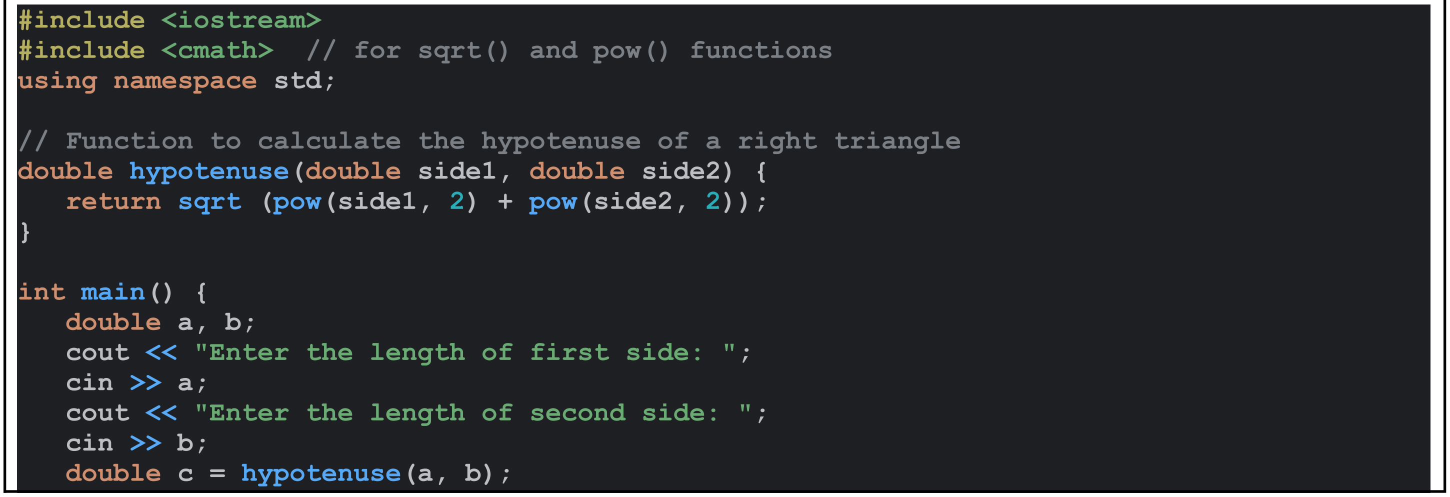
Output:



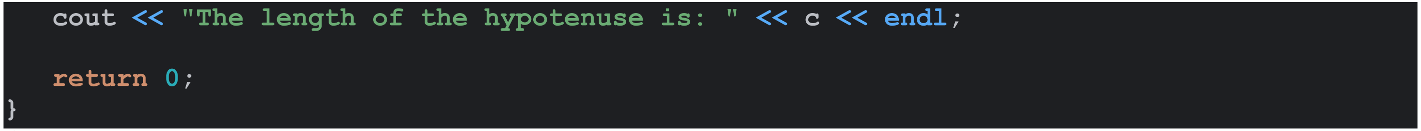
```
"C:\Users\Jaime Luis\CLionProjects\untitled2\cmake-build-debug\untitled2.exe"
Enter the side length of the cube:6


The volume of the cube is: 216


Process finished with exit code 0
```

2.

Code:

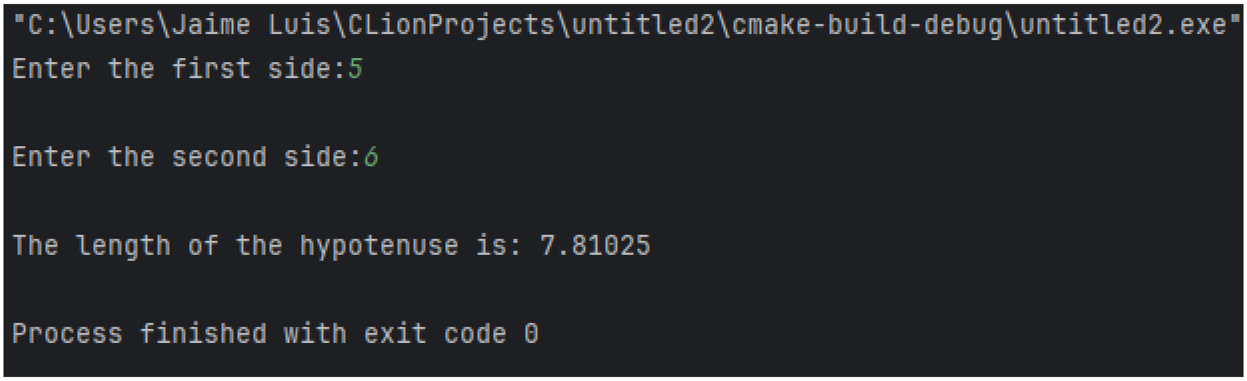


```cpp
#include <iostream>
#include <cmath>  // for sqrt() and pow() functions
using namespace std;

// Function to calculate the hypotenuse of a right triangle
double hypotenuse(double side1, double side2) {
   return sqrt (pow(side1, 2) + pow(side2, 2));
}

int main() {
   double a, b;
   cout << "Enter the length of first side: ";
   cin >> a;
   cout << "Enter the length of second side: ";
   cin >> b;
   double c = hypotenuse(a, b);
```



```
   cout << "The length of the hypotenuse is: " << c << endl;

   return 0;
}
```

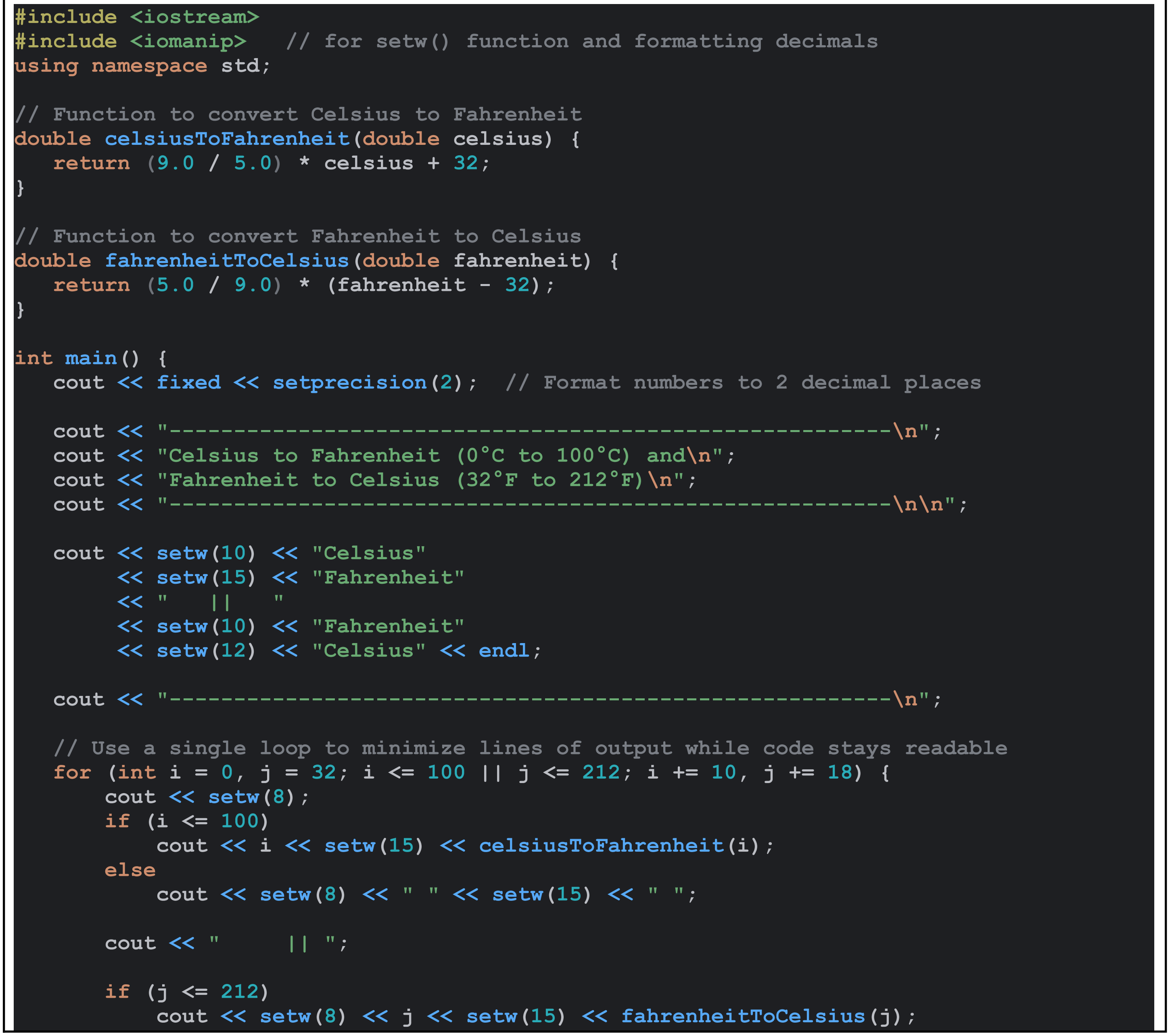
Output:



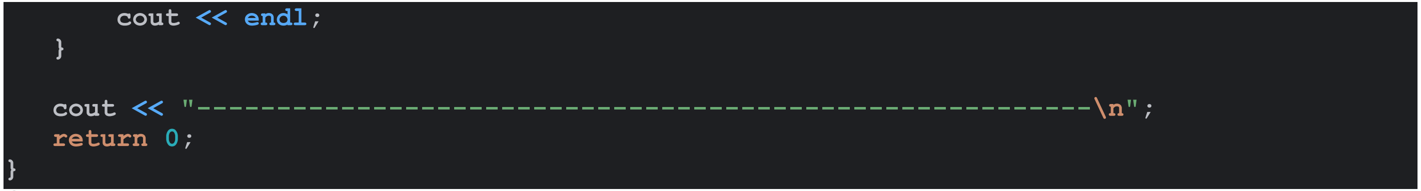
```
"C:\Users\Jaime Luis\CLionProjects\untitled2\cmake-build-debug\untitled2.exe"
Enter the first side:5


Enter the second side:6


The length of the hypotenuse is: 7.81025


Process finished with exit code 0
```

3.

Code:

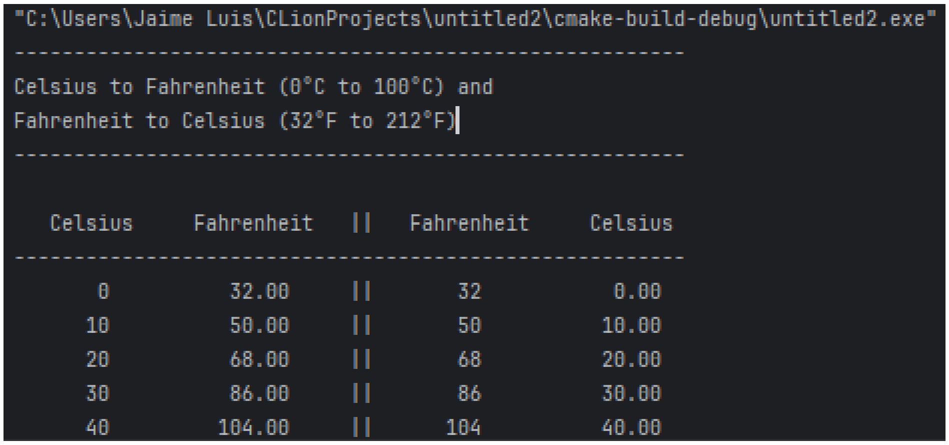


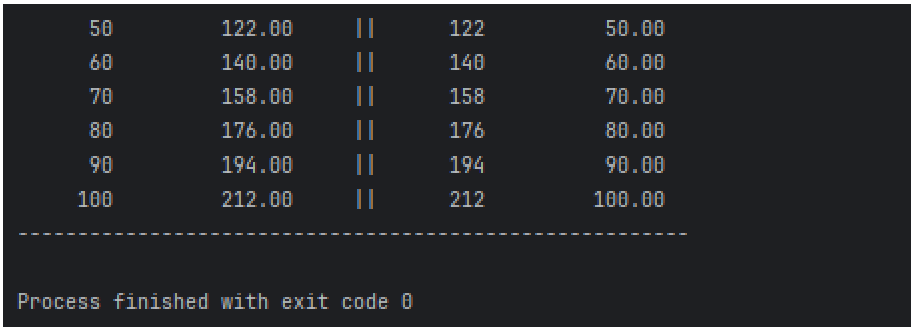
```
#include <iostream>
#include <iomanip>   // for setw() function and formatting decimals
using namespace std;

// Function to convert Celsius to Fahrenheit
double celsiusToFahrenheit(double celsius) {
   return (9.0 / 5.0) * celsius + 32;
}

// Function to convert Fahrenheit to Celsius
double fahrenheitToCelsius(double fahrenheit) {
   return (5.0 / 9.0) * (fahrenheit - 32);
}

int main() {
   cout << fixed << setprecision(2);  // Format numbers to 2 decimal places

   cout << "--------------------------------------------------------\n";
   cout << "Celsius to Fahrenheit (0°C to 100°C) and\n";
   cout << "Fahrenheit to Celsius (32°F to 212°F)\n";
   cout << "--------------------------------------------------------\n\n";

   cout << setw(10) << "Celsius"
        << setw(15) << "Fahrenheit"
        << "   ||   "
        << setw(10) << "Fahrenheit"
        << setw(12) << "Celsius" << endl;

   cout << "--------------------------------------------------------\n";

   // Use a single loop to minimize lines of output while code stays readable
   for (int i = 0, j = 32; i <= 100 || j <= 212; i += 10, j += 18) {
       cout << setw(8);
       if (i <= 100)
           cout << i << setw(15) << celsiusToFahrenheit(i);
       else
           cout << setw(8) << " " << setw(15) << " ";

       cout << "     || ";

       if (j <= 212)
           cout << setw(8) << j << setw(15) << fahrenheitToCelsius(j);
```



```
        cout << endl;
    }

    cout << "--------------------------------------------------------\n";
    return 0;
}
```

Output:





```
"C:\Users\Jaime Luis\CLionProjects\untitled2\cmake-build-debug\untitled2.exe"
------------------------------------------------------
Celsius to Fahrenheit (0°C to 100°C) and
Fahrenheit to Celsius (32°F to 212°F)
------------------------------------------------------


   Celsius     Fahrenheit   ||   Fahrenheit     Celsius
------------------------------------------------------
       0          32.00     ||       32           0.00
      10          50.00     ||       50          10.00
      20          68.00     ||       68          20.00
      30          86.00     ||       86          30.00
      40         104.00     ||      104          40.00
      50         122.00     ||      122          50.00
      60         140.00     ||      140          60.00
      70         158.00     ||      158          70.00
      80         176.00     ||      176          80.00
      90         194.00     ||      194          90.00
     100         212.00     ||      212         100.00
------------------------------------------------------


Process finished with exit code 0
```

7. Supplementary Activity

Analysis:

1. In this item, I programmed a computation for the volume of a cube using built-in mathematical functions. Before coding the main part, I coded a cmath header file so that I can access mathematical functions in C++. And then, I coded and initialized a function for the computation. I utilized a double data type because it is important that I print out the decimal values of an integer output. I returned the value through utilizing a pow function because the formula of the volume of the cube contains an exponent and pow function processes out the exponents. After the function, I will now initialize a main function for the inputting mechanism. I will declare a variable first and then I will now initialize a cin to let users input. After this, it will call out the function to process out the inputted value before printing out the final output.

2. In this item, I programmed a computation for the hypotenuse of a right triangle using the cmath mathematical function. Going through the same process, I coded a header file first to access C++ mathematical functions. Then, I coded a function for the computation. I utilized a double data type in the computation process. Two values will be inputted and then, pow function will be utilized first before proceeding to the sqrt function. The hypotenuse formula utilizes a pythagorean theorem and sqrt and pow functions mimic the pythagorean formula. After this, the value will now be returned. That's all for the function part. The main function would now be initialized for the inputting mechanism. The process would be the same as the first item except that there will be two values to be inputted and processed.

3. In this item, I programmed a table chart for the Fahrenheit equivalents of a Celsius temperature from 0 to 100 degrees, and the Celsius equivalents of all Fahrenheit temperatures from 32 to 212 degrees. I used an iomanip header for this item because I would no longer use a mathematical function and I will manipulate the output stream like the spacing for alignment and the decimals. After I coded the header file, I initialized a function for converting a celsius to fahrenheit and vice versa through coding its respective formulas. I initialized two functions for celsius-fahrenheit and fahrenheit-celsius conversions. The values would be returned based on its

formula then. Then, the main function would now be initialized to print out the table chart. The first line in the main function would be fixing the decimals into 2 decimals only using fixed object and setprecision () function. Then, the continuous broken lines would be printed out to make the output look like a table chart. It will be printed out 4 times in the whole output. After printing out the first continuous broken lines, the top message would be printed out and the second continuous broken lines would be printed out. Then, the celsius and fahrenheit referring text would be printed out so that the values would be correctly aligned to its temperature scales. I will now used a setw () function so that the text would be properly spaced and aligned. Through this function, there will be allotted preceding space before the actual text and the last text would be where the setw will stop from its starting point. And then, the third continuous broken line would now be printed out. I will use a for-loop statement to print out the multiple values and its converted one. There would be ten values for celsius-fahrenheit and fahrenheit-celsius conversion respectively. The setw () function would be printed out first for proper initial spacing before the values. Then, the unconverted one would be printed out. After this, there will be a setw again before printing out the converted one. After printing out the converted value, there will be a spacing before two vertical bars. A vertical bar would be used to divide the celsius-fahrenheit and fahrenheit-celsius conversion parts. After the vertical bar, the fahrenheit-celsius conversion part would now be printed out and it would still be the same process. The celsius-fahrenheit conversion is on the left side. After the for-loop whole process for the values, the fourth continuous broken line would be printed out and the whole coding would end.

8. Conclusion

In this activity, we are tasked to use a C++ built-in functions so that we can appreciate the C++ more and to minimize manual computations in addition to using separate functions previously. Before using the C++ built-in functions, a certain header files would need to be coded first. The header files that were used in this activity were cmath for mathematical functions and iomanip for spacing and decimal control. The first two items are when cmath header files were used and its functions. They are also items when the built-in functions were placed in the separation function and not in the main function. It is because they would calculate for the algebraic values, thus built-in mathematical functions were crucial for the process. And in the third item, the table chart would be programmed. Thus, the iomanip header file was utilized. The built-in functions were now placed in the main function. The setprecision was used to set a limit on how many decimals would be printed out and setw is for the spacing and alignment. In conclusion, using built-in functions would make us understand our code more because of the functions name and make our code more readable unlike in manual computations.