Activity No. 6.2	
Built-in Functions	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed:10/24/25**
Section:CPE11S1	**Date Submitted:10/24/25**
Name(s): Will Stuart D. Ponce Jr.	**Instructor:** Engr. Jimlord M. Quejado

6.Activities:

Note: Perform each of these steps for the below problems:

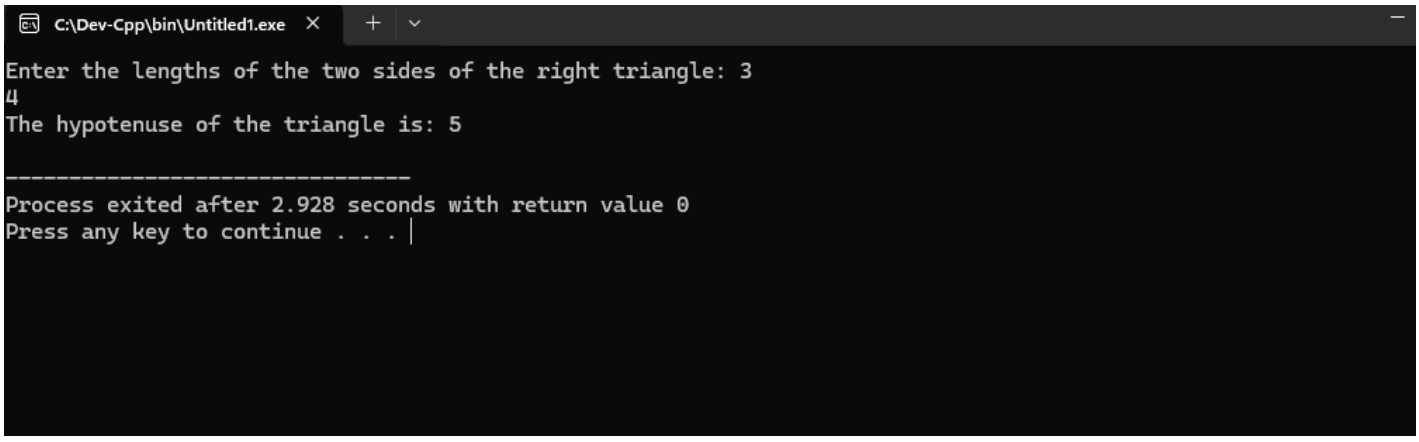
Read the problem statement
Write a C program
Test, Debug and execute the C program
Create a screenshot of the code and output and submit it using the activity template

Create a program that defines a function to compute for the volume of a cube. The formula of the volume of the cube is given as V = s * s * s.
Define a function hypotenuse that calculates the length of the hypotenuse of a right triangle when the other two sides are given. Use this function in a program to determine the length of the hypotenuse for each of the following triangles. The function takes two arguments of type double and returns the hypotenuse as a double.
Implement the following integer functions:
Function celsius returns the Celsius equivalent of a Fahrenheit temperature.
Function fahrenheit returns the Fahrenheit equivalent of a Celsius temperature.
Use these functions to write a program that prints charts showing the Fahrenheit equivalents of a Celsius temperatures from 0 to 100 degrees, and the Celsius equivalents of all Fahrenheit temperatures from 32 to 212 degrees. Print the outputs in a neat tabular format that minimizes the number of lines of output while remaining readable.

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

// Function prototype
double cubeVolume(double side);

int main() {
    double side, volume;

    cout << "Enter the side length of the cube: ";
    cin >> side;

    volume = cubeVolume(side);

    cout << "The volume of the cube with side " << side
         << " is: " << volume << endl;

    return 0;
}

// Function definition
double cubeVolume(double side) {
    return side * side * side;
}
```

```
Enter the side length of the cube: 3
The volume of the cube with side 3 is: 27

--------------------------------
Process exited after 2.956 seconds with return value 0
Press any key to continue . . .
```

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

// Function prototype
double hypotenuse(double side1, double side2);

int main() {
    double a, b, h;

    cout << "Enter the lengths of the two sides of the right triangle: ";
    cin >> a >> b;

    h = hypotenuse(a, b);

    cout << "The hypotenuse of the triangle is: " << h << endl;

    return 0;
}

// Function definition
double hypotenuse(double side1, double side2) {
    return sqrt(side1 * side1 + side2 * side2);
}
```



```cpp
#include <iostream>
#include <iomanip> // for setw() and setprecision()
using namespace std;

// Function prototypes
double celsius(double fahrenheit);
double fahrenheit(double celsius);

int main() {
    cout << fixed << setprecision(2);

    cout << "Celsius to Fahrenheit Conversion Table\n";
    cout << "--------------------------------------\n";
    cout << setw(10) << "Celsius" << setw(15) << "Fahrenheit" << endl;

    for (int c = 0; c <= 100; c += 10) {
        cout << setw(10) << c << setw(15) << fahrenheit(c) << endl;
    }

    cout << "\nFahrenheit to Celsius Conversion Table\n";
    cout << "--------------------------------------\n";
    cout << setw(12) << "Fahrenheit" << setw(15) << "Celsius" << endl;

    for (int f = 32; f <= 212; f += 10) {
        cout << setw(12) << f << setw(15) << celsius(f) << endl;
    }

    return 0;
}

// Function definitions
double celsius(double fahrenheit) {
    return (5.0 / 9.0) * (fahrenheit - 32);
}
```

```
Celsius to Fahrenheit Conversion Table
--------------------------------------
   Celsius     Fahrenheit
         0          32.00
        10          50.00
        20          68.00
        30          86.00
        40         104.00
        50         122.00
        60         140.00
        70         158.00
        80         176.00
        90         194.00
       100         212.00

Fahrenheit to Celsius Conversion Table
--------------------------------------
  Fahrenheit        Celsius
          32           0.00
          42           5.56
          52          11.11
          62          16.67
          72          22.22
          82          27.78
          92          33.33
         102          38.89
         112          44.44
         122          50.00
         132          55.56
         142          61.11
         152          66.67
         162          72.22
         172          77.78
         182          83.33
         192          88.89
         202          94.44
         212         100.00

--------------------------------
```

7.Supplementary Activity

Analysis:In this program, the goal was to calculate the volume of a cube using a user-defined function. The program first asks the user to input the side length of the cube, then calls a function named cubeVolume() that computes the volume using the formula $V=s\times s\times s$
V=s×s×s. The result is then displayed on the screen with the input value. This activity helps in understanding how to create and call functions in C++, as well as how to pass arguments and return values.

By separating the calculation into a function, the code becomes more organized and reusable. The use of the double data type allows for precise results, even for decimal inputs. Overall, this simple program demonstrates how to perform basic arithmetic operations, handle user input and output, and structure a program using function prototypes — which are important concepts in modular programming.

Analysis: This program calculates the hypotenuse of a right triangle using a custom function called hypotenuse(). The user is prompted to enter the lengths of the two shorter sides (often referred to as the base and height). The function then applies the Pythagorean theorem $h=a2+b2$h=a2+b2
to find the length of the hypotenuse. The <cmath> library is used for the sqrt() function, which performs the square root operation.

The program illustrates how mathematical formulas can be implemented in C++ using functions and standard libraries. By defining the logic in a separate function, the program becomes easier to understand and maintain. It also reinforces the use of function prototypes, data handling with double variables, and output formatting. This exercise provides practical experience in applying mathematical concepts to programming.

Analysis: In this activity, the program uses two functions — celsius() and fahrenheit() — to convert temperatures between Celsius and Fahrenheit. The first function converts Fahrenheit to Celsius using the formula $C=59(F-32)$C=95(F−32), while the second converts Celsius to Fahrenheit with $F=95C+32$F=59

C+32. The program then prints two neatly formatted tables: one showing Celsius-to-Fahrenheit conversions from 0 to 100 degrees, and another showing Fahrenheit-to-Celsius conversions from 32 to 212 degrees.
This program demonstrates the importance of function usage for code clarity and reusability. It also introduces formatted output using the <iomanip> library for aligned tables and controlled decimal precision. Through this exercise, the student learns how to apply real-world formulas in programming, manage loops effectively, and present data in an organized and readable format. Overall, it combines both logic and presentation skills in C++ programming.

8. Conclusion

Through the completion of these three C++ programs, I was able to gain a deeper understanding of how functions work and why they are essential in structuring a program. Each task emphasized the importance of breaking down a problem into smaller, reusable parts. The cube volume program introduced the idea of using a simple function to perform mathematical operations, while the hypotenuse and temperature conversion programs showed how multiple functions can make a program more modular and organized. This helped me appreciate how function prototypes and definitions improve program readability and maintainability.

In addition, I learned how to use input and output statements efficiently, how to apply mathematical formulas in programming, and how to handle real numbers using the double data type. The exercises also enhanced my knowledge of standard C++ libraries such as <cmath> for mathematical operations and <iomanip> for formatted output. These libraries made it easier to write clean and professional-looking programs that are both functional and user-friendly.

Overall, this activity was a great opportunity to apply both logical thinking and coding skills to solve real-world problems. It strengthened my understanding of fundamental programming concepts such as modularity, data handling, and arithmetic computation. By writing, testing, and debugging each program, I developed more confidence in using C++ for problem-solving and gained a better appreciation for the importance of structured programming in software development.