Activity No. 1.1

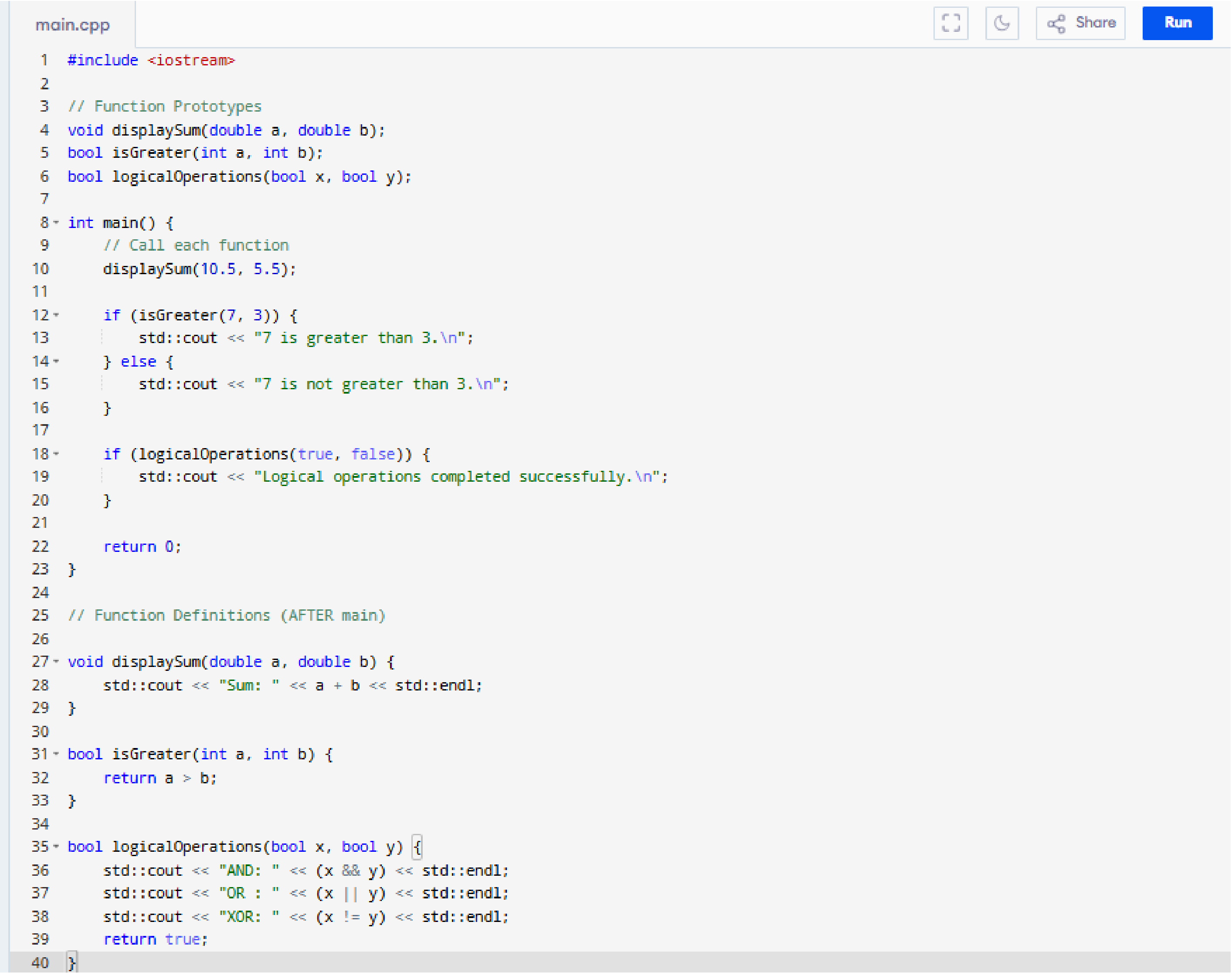
Hands-on Activity 1.1 Basic C++ Programming

Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 7/30/25**
Section: CPE21S4	**Date Submitted: 7/31/25**
Name(s): Avila, Vince Gabriel V.	**Instructor: Jimlord Quejado**

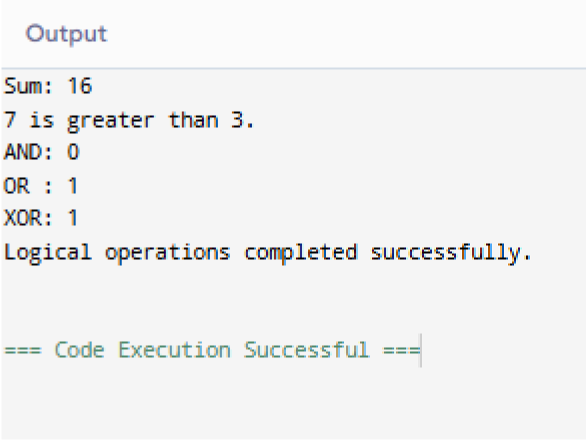
6. Procedure

ILO A: Create Code That Follows the Basic C++ Code Structure

OUTPUT:



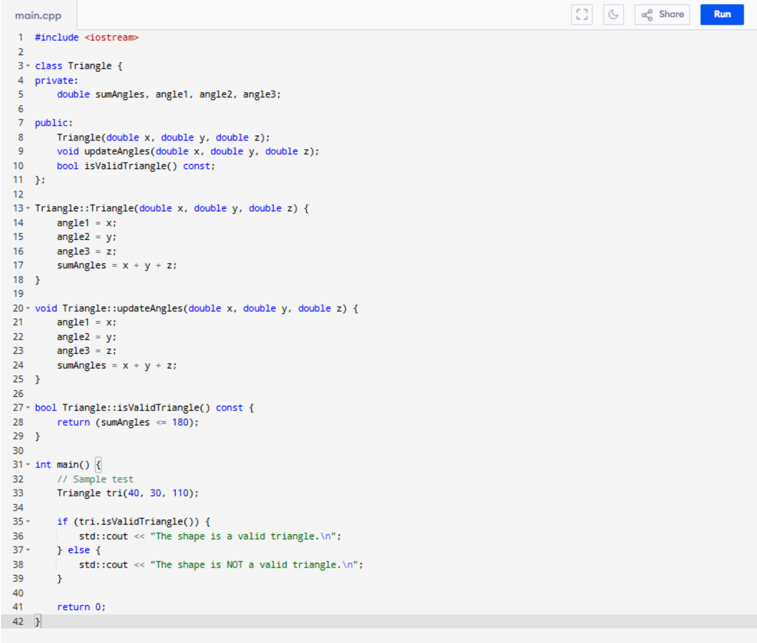
```cpp
#include <iostream>

// Function Prototypes
void displaySum(double a, double b);
bool isGreater(int a, int b);
bool logicalOperations(bool x, bool y);

int main() {
    // Call each function
    displaySum(10.5, 5.5);

    if (isGreater(7, 3)) {
        std::cout << "7 is greater than 3.\n";
    } else {
        std::cout << "7 is not greater than 3.\n";
    }

    if (logicalOperations(true, false)) {
        std::cout << "Logical operations completed successfully.\n";
    }

    return 0;
}

// Function Definitions (AFTER main)

void displaySum(double a, double b) {
    std::cout << "Sum: " << a + b << std::endl;
}

bool isGreater(int a, int b) {
    return a > b;
}

bool logicalOperations(bool x, bool y) {
    std::cout << "AND: " << (x && y) << std::endl;
    std::cout << "OR : " << (x || y) << std::endl;
    std::cout << "XOR: " << (x != y) << std::endl;
    return true;
}
```



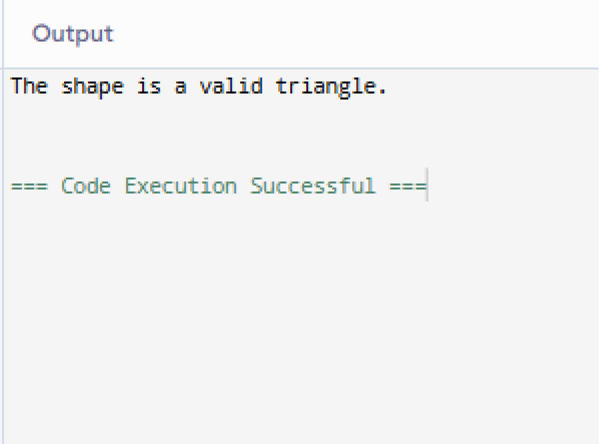
```
Sum: 16
7 is greater than 3.
AND: 0
OR : 1
XOR: 1
Logical operations completed successfully.


=== Code Execution Successful ===
```

ILO B: Implement Appropriate Class Definition and Instances Based on Given Requirements

OUTPUT:



```cpp
#include <iostream>

class Triangle {
private:
    double sumAngles, angle1, angle2, angle3;

public:
    Triangle(double x, double y, double z);
    void updateAngles(double x, double y, double z);
    bool isValidTriangle() const;
};

Triangle::Triangle(double x, double y, double z) {
    angle1 = x;
    angle2 = y;
    angle3 = z;
    sumAngles = x + y + z;
}

void Triangle::updateAngles(double x, double y, double z) {
    angle1 = x;
    angle2 = y;
    angle3 = z;
    sumAngles = x + y + z;
}

bool Triangle::isValidTriangle() const {
    return (sumAngles <= 180);
}

int main() {
    // Sample test
    Triangle tri(40, 30, 110);

    if (tri.isValidTriangle()) {
        std::cout << "The shape is a valid triangle.\n";
    } else {
        std::cout << "The shape is NOT a valid triangle.\n";
    }

    return 0;
}
```

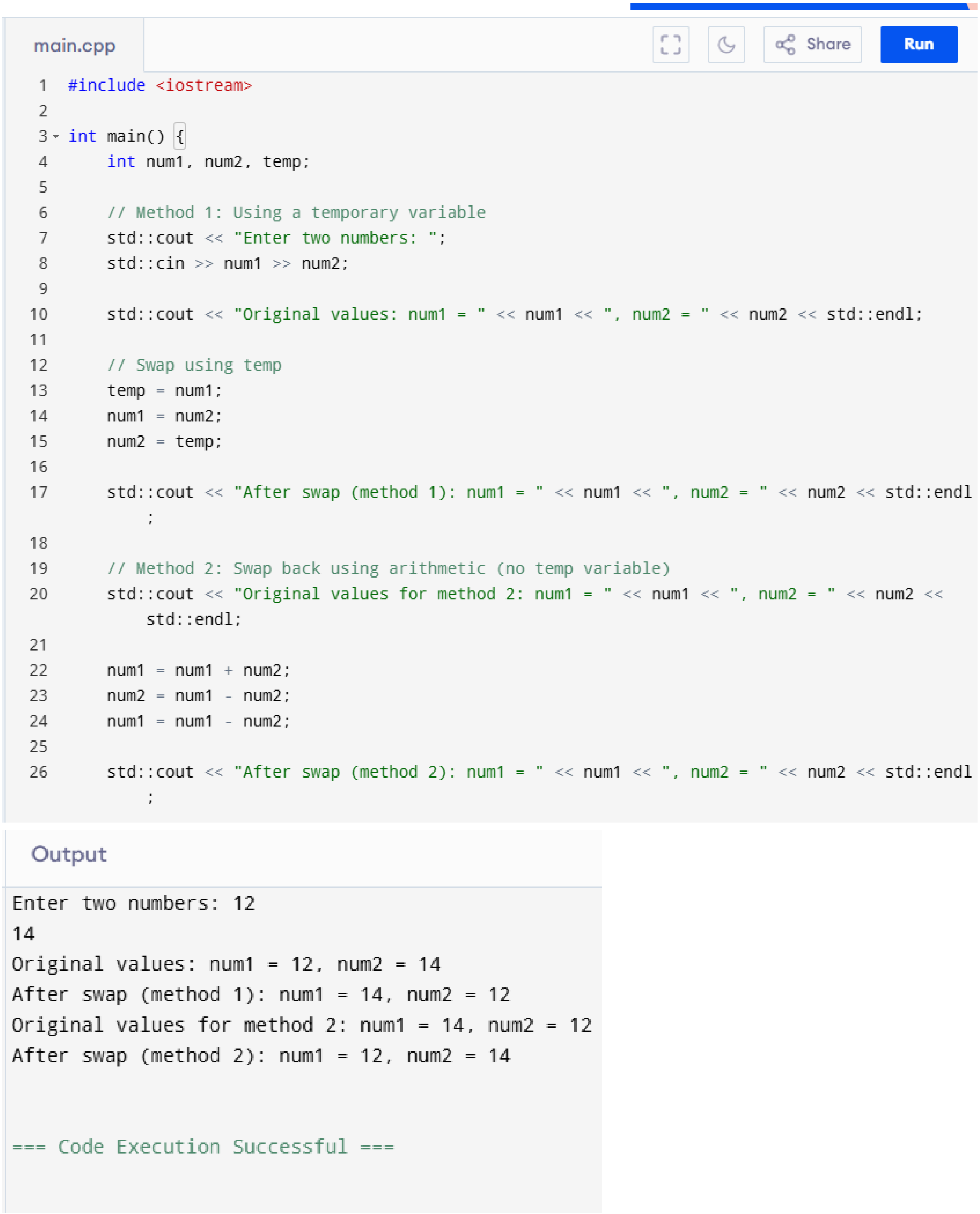


```
The shape is a valid triangle.


=== Code Execution Successful ===
```

7. Supplementary Activity

ILO C: Solve Different Problems using the C++ Programming Language

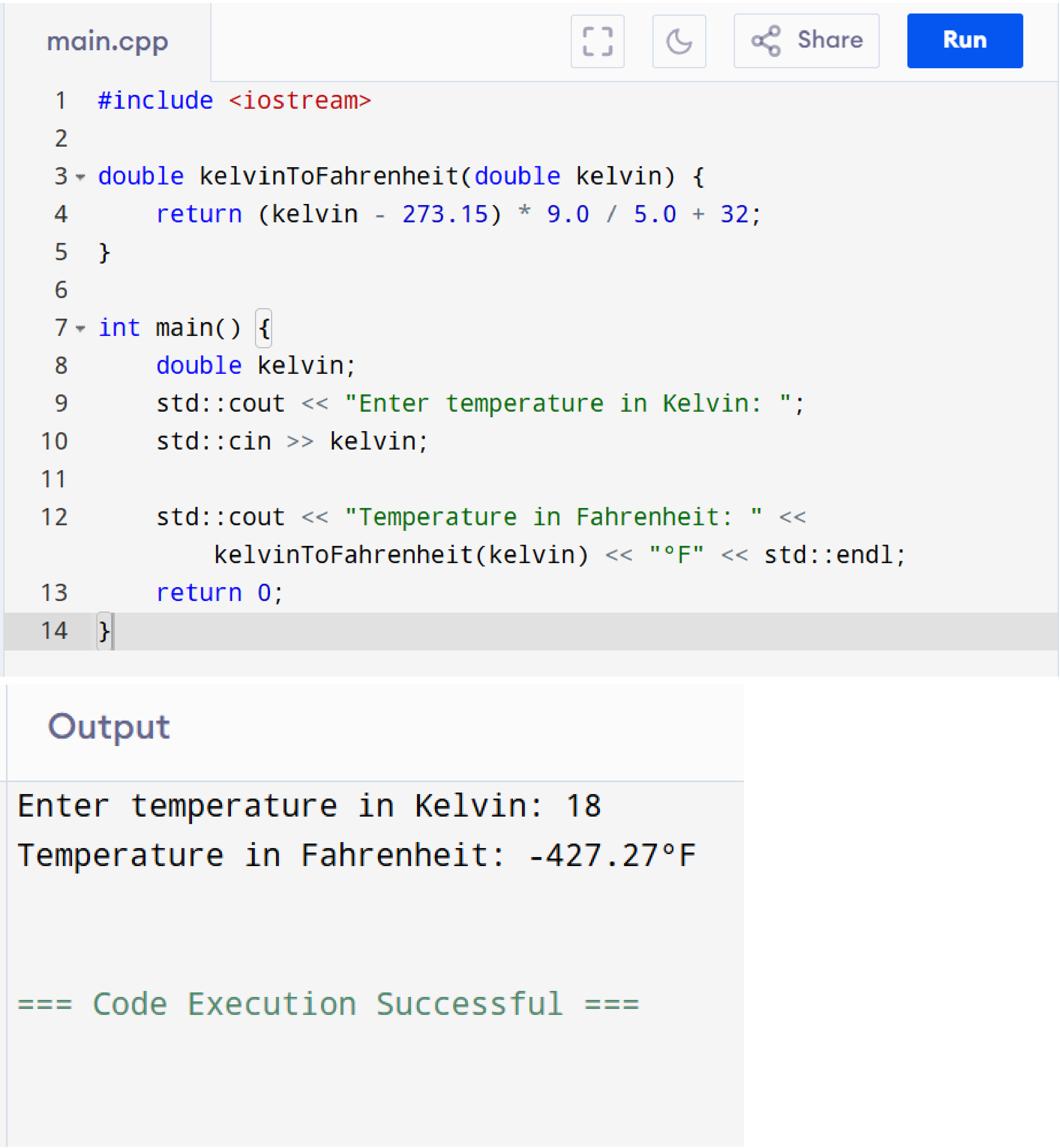
1. Create a C++ program to swap the two numbers in different variables



```cpp
#include <iostream>

int main() {
    int num1, num2, temp;

    // Method 1: Using a temporary variable
    std::cout << "Enter two numbers: ";
    std::cin >> num1 >> num2;

    std::cout << "Original values: num1 = " << num1 << ", num2 = " << num2 << std::endl;

    // Swap using temp
    temp = num1;
    num1 = num2;
    num2 = temp;

    std::cout << "After swap (method 1): num1 = " << num1 << ", num2 = " << num2 << std::endl;

    // Method 2: Swap back using arithmetic (no temp variable)
    std::cout << "Original values for method 2: num1 = " << num1 << ", num2 = " << num2 << std::endl;

    num1 = num1 + num2;
    num2 = num1 - num2;
    num1 = num1 - num2;

    std::cout << "After swap (method 2): num1 = " << num1 << ", num2 = " << num2 << std::endl;
```

```
Enter two numbers: 12
14
Original values: num1 = 12, num2 = 14
After swap (method 1): num1 = 14, num2 = 12
Original values for method 2: num1 = 14, num2 = 12
After swap (method 2): num1 = 12, num2 = 14


=== Code Execution Successful ===
```

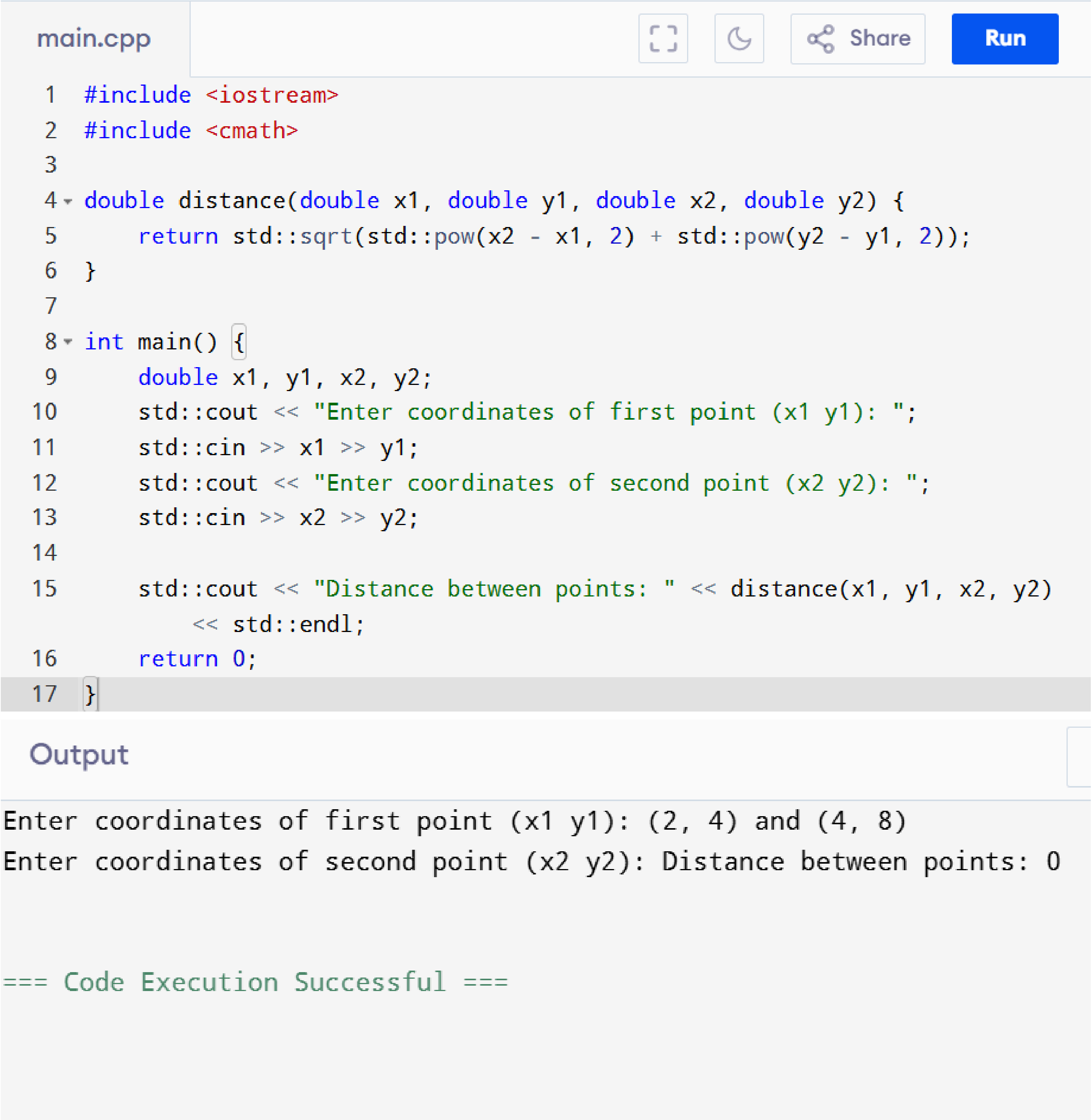
2. Create a C++ program that has a function to convert temperature in Kelvin to Fahrenheit.



```cpp
#include <iostream>

double kelvinToFahrenheit(double kelvin) {
    return (kelvin - 273.15) * 9.0 / 5.0 + 32;
}

int main() {
    double kelvin;
    std::cout << "Enter temperature in Kelvin: ";
    std::cin >> kelvin;

    std::cout << "Temperature in Fahrenheit: " <<
        kelvinToFahrenheit(kelvin) << "°F" << std::endl;
    return 0;
}
```

```
Enter temperature in Kelvin: 18
Temperature in Fahrenheit: -427.27°F



=== Code Execution Successful ===
```

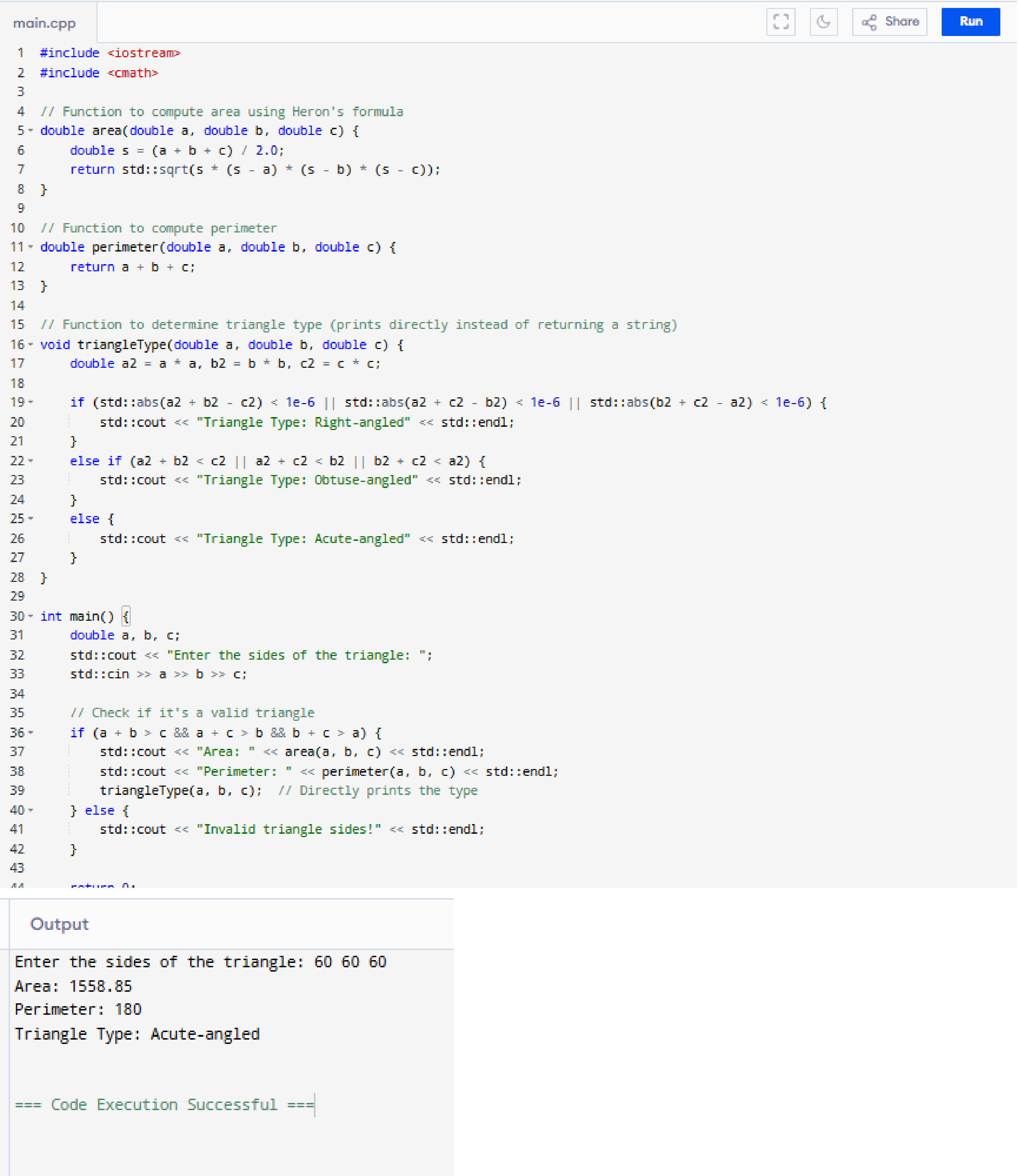
3. Create a C++ program that has a function that will calculate the distance between two points.



```cpp
#include <iostream>
#include <cmath>

double distance(double x1, double y1, double x2, double y2) {
    return std::sqrt(std::pow(x2 - x1, 2) + std::pow(y2 - y1, 2));
}

int main() {
    double x1, y1, x2, y2;
    std::cout << "Enter coordinates of first point (x1 y1): ";
    std::cin >> x1 >> y1;
    std::cout << "Enter coordinates of second point (x2 y2): ";
    std::cin >> x2 >> y2;

    std::cout << "Distance between points: " << distance(x1, y1, x2, y2)
        << std::endl;
    return 0;
}
```

```
Enter coordinates of first point (x1 y1): (2, 4) and (4, 8)
Enter coordinates of second point (x2 y2): Distance between points: 0


=== Code Execution Successful ===
```

4. Modify the code given in ILO B and add the following functions:
a. A function to compute for the area of a triangle
b. A function to compute for the perimeter of a triangle
c. A function that determines whether the triangle is acute-angled, obtuse-angled or ‘others.’



```cpp
#include <iostream>
#include <cmath>

// Function to compute area using Heron's formula
double area(double a, double b, double c) {
    double s = (a + b + c) / 2.0;
    return std::sqrt(s * (s - a) * (s - b) * (s - c));
}

// Function to compute perimeter
double perimeter(double a, double b, double c) {
    return a + b + c;
}

// Function to determine triangle type (prints directly instead of returning a string)
void triangleType(double a, double b, double c) {
    double a2 = a * a, b2 = b * b, c2 = c * c;

    if (std::abs(a2 + b2 - c2) < 1e-6 || std::abs(a2 + c2 - b2) < 1e-6 || std::abs(b2 + c2 - a2) < 1e-6) {
        std::cout << "Triangle Type: Right-angled" << std::endl;
    }
    else if (a2 + b2 < c2 || a2 + c2 < b2 || b2 + c2 < a2) {
        std::cout << "Triangle Type: Obtuse-angled" << std::endl;
    }
    else {
        std::cout << "Triangle Type: Acute-angled" << std::endl;
    }
}

int main() {
    double a, b, c;
    std::cout << "Enter the sides of the triangle: ";
    std::cin >> a >> b >> c;

    // Check if it's a valid triangle
    if (a + b > c && a + c > b && b + c > a) {
        std::cout << "Area: " << area(a, b, c) << std::endl;
        std::cout << "Perimeter: " << perimeter(a, b, c) << std::endl;
        triangleType(a, b, c);  // Directly prints the type
    } else {
        std::cout << "Invalid triangle sides!" << std::endl;
    }

    return 0;
```

```
Enter the sides of the triangle: 60 60 60
Area: 1558.85
Perimeter: 180
Triangle Type: Acute-angled


=== Code Execution Successful ===
```

8. Conclusion

Provide the following:

- Summary of lessons learned
  This activity recalls the fundamentals of c++ and I realise how the structure of the code is clearly defined in the struggle to organize my code like global declarations and main functions.
- Analysis of the procedure

The activity is tough when you don't have some background knowledge of coding. It was particularly tough for me. Even simple tasks like computing checking triangle properties pushed me to be precise in logic

- Analysis of the supplementary activity
  The supplementary task kinda looked simple but it really requires real focus to it. The logic challenges me to be more detail oriented.
- Concluding statement / Feedback: How well did you think you did in this activity? What are your areas for improvement?
  To be honest it was pretty tough for me. I struggled with both logic and remembering how to use certain c++.

9. Assessment Rubric