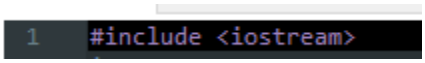
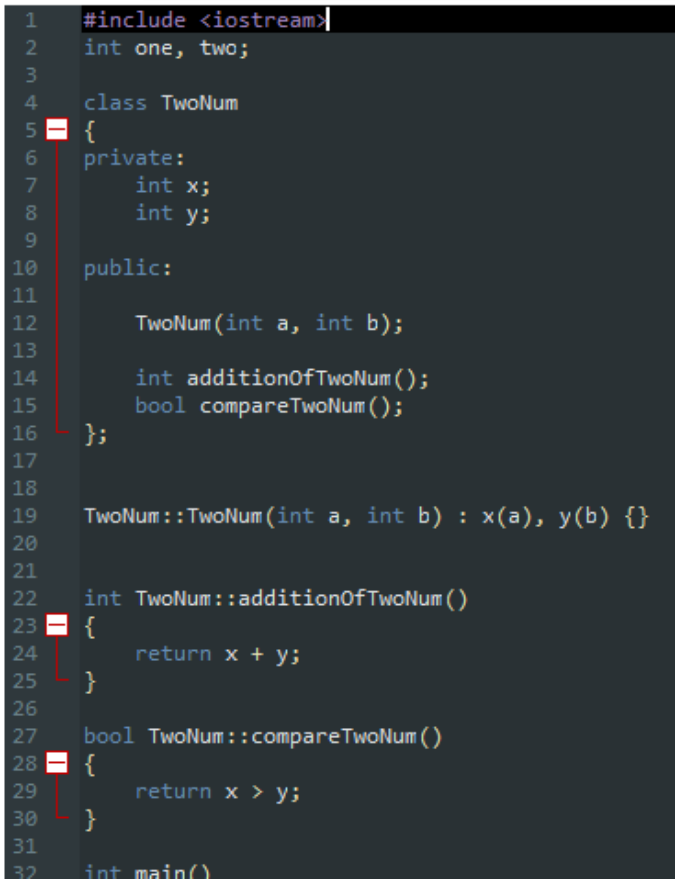
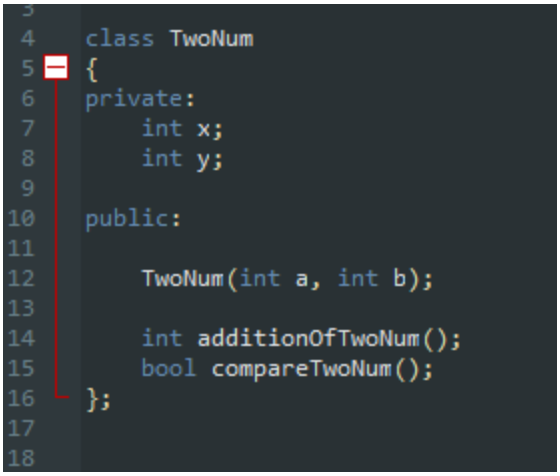
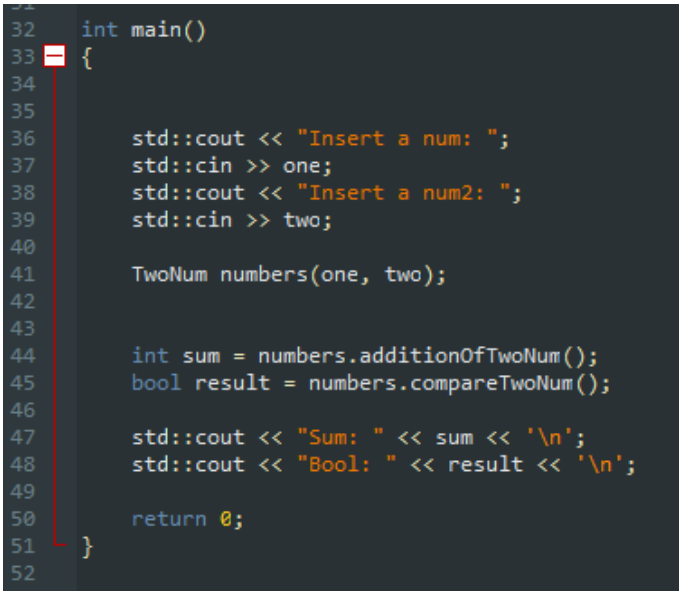
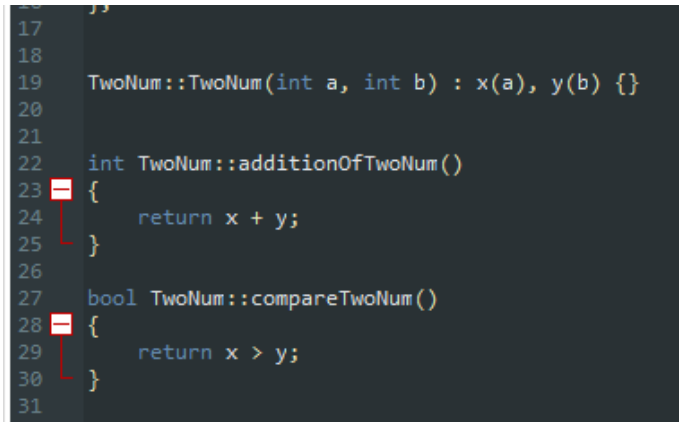
Activity No. 1	
Hands-on Activity 1.1 Basic C++ Programming	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 09/09/24**
Section: CPE21S4	**Date Submitted: 09/10/24**
Name(s): Pornobe, Reuel Christian, M.	**Instructor: Prof. Rizette Sayo**
6. Output	

Sections	Answer
Header File Declaration Section	
Global Declaration Section	

Class Declaration and Method Definition Section	
Main Function	
Method Definition	

```
class TwoNum
{
private:
    int x;
    int y;

public:

    TwoNum(int a, int b);

    int additionOfTwoNum();
    bool compareTwoNum();
};
```

```
int main()
{


    std::cout << "Insert a num: ";
    std::cin >> one;
    std::cout << "Insert a num2: ";
    std::cin >> two;

    TwoNum numbers(one, two);


    int sum = numbers.additionOfTwoNum();
    bool result = numbers.compareTwoNum();

    std::cout << "Sum: " << sum << '\n';
    std::cout << "Bool: " << result << '\n';

    return 0;
}
```

```


TwoNum::TwoNum(int a, int b) : x(a), y(b) {}


int TwoNum::additionOfTwoNum()
{
    return x + y;
}

bool TwoNum::compareTwoNum()
{
    return x > y;
}
```

B.

```cpp
#include <iostream>


class Triangle                                  //Class constructor
{
private:                                        //Class Attributes
double totalAngle, angleA, angleB, angleC;
public:
Triangle(double A, double B, double C);
void setAngles(double A, double B, double C);
const bool validateTriangle();
};

Triangle::Triangle(double A, double B, double C)  //Class Method Definition
{
angleA = A;
angleB = B;
angleC = C;
totalAngle = A+B+C;
}

void Triangle::setAngles(double A, double B, double C) //Class Method Definition
{
angleA = A;
angleB = B;
angleC = C;
totalAngle = A+B+C;
}

const bool Triangle::validateTriangle()  //Class Method Definition
{
return (totalAngle <= 180);
}

int main()       //Main Function
{

Triangle set1(40, 30, 110);       //Calling class
if(set1.validateTriangle()){      // Calling a class function
std::cout << "The shape is a valid triangle.\n";
} else {
std::cout << "The shape is NOT a valid triangle.\n";
}
return 0;
}

```

```
C:\Users\Reuel\Desktop\tryretry.exe
The shape is a valid triangle.

--------------------------------
Process exited after 0.05468 seconds w
Press any key to continue . . .
```

7. Supplementary Activity

1. Swap two Numbers:

```cpp
#include <iostream>
int a, b, c;

int swapping(int a, int b)
{
    c = a;
    a = b;
    b = c;

    std::cout << "After the swap: " << '\n' <<  "Number 1: " << a << " " << "Number 2: " << b << '\n';
    return 0;
}

int main()
{
    std::cout << "Insert a number: ";
    std:: cin >> a;
    std::cout << "Insert a second number: ";
    std::cin >> b;

    std::cout << "Before the swap: " << '\n' <<  "Number 1: " << a << " " << "Number 2: " << b << '\n';
    swapping(a,b);

    return 0;
}
```

```
C:\Users\Reuel\Desktop\1 1.2.exe
Insert a number: 1
Insert a second number: 2
Before the swap:
Number 1: 1 Number 2: 2
After the swap:
Number 1: 2 Number 2: 1

--------------------------------
Process exited after 6.927 seconds with return
Press any key to continue . . .
```

2.Kelvin to Fahrenheit

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


double kelvinConverter(double kelvin)
{
    return (kelvin - 273.15) * 1.8 + 32.0;
}

int main()
{
    cout << "Enter the kelvin to be converted: ";
    cin >> kelvin;

    double fahrenheit = kelvinConverter(kelvin);
    cout << "Temperature in Fahrenheit: " << fahrenheit << '\n';

    return 0;
}
```

```
C:\Users\Reuel\Desktop\asdasd.exe
Enter the kelvin to be converted: 100
Temperature in Fahrenheit: -279.67

--------------------------------
Process exited after 2.032 seconds with return value 0
Press any key to continue . . .
```

3. Distance between two points

[*] 1.2 A.cpp | 1 1.2.cpp | asdasd.cpp | hahahahah.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;
double xPointOne, yPointOne, xPointTwo, yPointTwo;



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



int main() {


    cout << "Enter x1: ";
    cin >> xPointOne;
    cout << "Enter y1: ";
    cin >> yPointOne;
    cout << "Enter x2: ";
    cin >> xPointTwo;
    cout << "Enter y2: ";
    cin >> yPointTwo;

    double distance = distanceBetweenTwoPoints(xPointOne, yPointOne, xPointTwo, yPointTwo);
    cout << "Distance between the points: " << distance << '\n';

    return 0;
}
```

```
C:\Users\Reuel\Desktop\hahahahah.exe
Enter x1: 1
Enter y1: 2
Enter x2: 3
Enter y2: 4
Distance between the points: 2.82843

--------------------------------
Process exited after 3.356 seconds with return v
Press any key to continue . . .
```

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>
#include <string>
using namespace std;

double angle1, angle2, angle3, side1, side2, side3;

class Triangle
{
private:
    double angle1, angle2, angle3;
    double side1, side2, side3;

public:
    Triangle(double angle1, double angle2, double angle3, double side1, double side2, double side3);
    void updateAngles(double angle1, double angle2, double angle3);
    void updateSides(double side1, double side2, double side3);
    bool isValidTriangle();
    double calculateArea();
    double calculatePerimeter();
    string determineTriangleType();
};

Triangle::Triangle(double angle1, double angle2, double angle3, double side1, double side2, double side3)
    : angle1(angle1), angle2(angle2), angle3(angle3), side1(side1), side2(side2), side3(side3) {}

void Triangle::updateAngles(double angle1, double angle2, double angle3)
{
    this->angle1 = angle1;
    this->angle2 = angle2;
    this->angle3 = angle3;
}

void Triangle::updateSides(double side1, double side2, double side3)
{
    this->side1 = side1;
    this->side2 = side2;
    this->side3 = side3;
}

bool Triangle::isValidTriangle()
{

    return (angle1 + angle2 + angle3 == 180) && (side1 > 0 && side2 > 0 && side3 > 0);
}

double Triangle::calculateArea() //a. A function to compute for the area of a triangle
{

    double s = calculatePerimeter() / 2;
    return sqrt(s * (s - side1) * (s - side2) * (s - side3));
}

double Triangle::calculatePerimeter() //b. A function to compute for the perimeter of a triangle
{
    return side1 + side2 + side3;
}

string Triangle::determineTriangleType() //c. A function that determines whether the triangle is acute-angled, obtuse-angled or ‘others.’
{
    if (!isValidTriangle()) return "Invalid";

    if (angle1 < 90 && angle2 < 90 && angle3 < 90) return "Acute-angled";
    if (angle1 == 90 || angle2 == 90 || angle3 == 90) return "Right-angled";
    return "Obtuse-angled";
}
```

```
C:\Users\Reuel\Desktop\treyrteytr.exe
Enter the three angles of the triangle (in degrees):
40
50
90
Enter the three sides of the triangle:
5
12
13
The shape is a valid triangle.
Area: 30
Perimeter: 30
Type: Right-angled

--------------------------------
Process exited after 20.98 seconds with return value 0
Press any key to continue . . .
```

```cpp
int main() {
    cout << "Enter the three angles of the triangle (in degrees):\n";
    cin >> angle1 >> angle2 >> angle3;
    cout << "Enter the three sides of the triangle:\n";
    cin >> side1 >> side2 >> side3;


    Triangle userTriangle(angle1, angle2, angle3, side1, side2, side3);

    if(userTriangle.isValidTriangle()) {
        cout << "The shape is a valid triangle.\n";
        cout << "Area: " << userTriangle.calculateArea() << "\n";
        cout << "Perimeter: " << userTriangle.calculatePerimeter() << "\n";
        cout << "Type: " << userTriangle.determineTriangleType() << "\n";
    } else {
        cout << "The shape is NOT a valid triangle.\n";
    }

    return 0;
}
```

8. Conclusion

For this Lab Activity, I learned about the basic C++ code structure. I learned about syntax and proper construction. Furthermore, I learned how to create classes and appropriate arguments in C++. I was able to create a class with arguments in C++. I was able to solve problems given at hand. I learned the use of class and how it is detrimental in coding. Class helps in creating an organized and structured data that can be easily called when needed in the program.

In the first part of the program, we declare the header files and the global variables needed. After that, Add the class and its attributes. Then, add the method functions needed or required by the program. Lastly, the main function is where we call the class. After that, we call the functions of the class to send out the output of the program.

The supplementary activity strengthened my knowledge about C++ and its syntax. It reinforced the uses and syntax of the user-defined functions(in 1,2 , and 3). For the number 4, it built up further my understanding of class constructors and class functions.

In this activity, I did well in understanding the concepts and structures of classes in C++. I was able to fully utilize the class and its function in the program. However, I think I need improvements in learning the syntax of C++. The coding took a lot of time since I had to recheck my notes about the syntax of C++.

9. Assessment Rubric