Activity No. <n>	
<Replace with Title>	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 7/29/2025**
Section: CPE21S4	**Date Submitted: 8/7/2025**
Name(s): Anastacio, Lester Arvid P.	**Instructor: Engr. Jimlord M. Quejado**

6. Output

Procedure:

Step 1- 2:

#include <iostream> Untitled-1

```
#include <iostream>

class Triangle{
    private:
    double totalAngle, angleA, angleB, angle C;
}
```

Step 3:

```
#include <iostream>

class Triangle{
    private:
        double totalAngle, angleA, angleB, angle C;

    public:
        Triangle(double A, double B, double C);
        void setAngles(double A, double B, double C);
        const bool validateTriangle();
};
```

Step 4:

```cpp
#include <iostream>

class Triangle{
    private:
        double totalAngle, angleA, angleB, angle C;

    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){
            angleA = A;
            angleB = B;
            angleC = C;
            totalAngle = A+B+C;
    }

        void Triangle::setAngles(double A, double B, double C){
            angleA = A;
            angleB = B;
            angleC = C;
            totalAngle = A+B+C;

        }

        const bool Triangle::validateTriangle(){
            return (totalAngle <= 180);
        }
```

Step 5:

```cpp
int main(){
    //driver code
    Triangle set1(40, 30, 110);
    if(set1.validateTriangle()){
        std::cout << "The shape is a valid triangle.\n";
    }
    return 0;
}
```

Step 6: Output

```
Output

The shape is a valid triangle.


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

Observations:
I've found that it the code checks if the Triangle is indeed a triangle with the use of angles it checks if the total sum amount of the values is 180.

7. Supplementary Activity

1. Swapping two numbers

CODE:

```cpp
#include <iostream>

void swapNumbers(int& a, int& b) {
    int temp = a;
    a = b;
    b = temp;
}

int main() {
    int x = 5, y = 10;
    std::cout << "Before swap: x = " << x << ", y = " << y << std::endl;
    swapNumbers(x, y);
    std::cout << "After swap: x = " << x << ", y = " << y << std::endl;
    return 0;
}
```

OUTPUT:

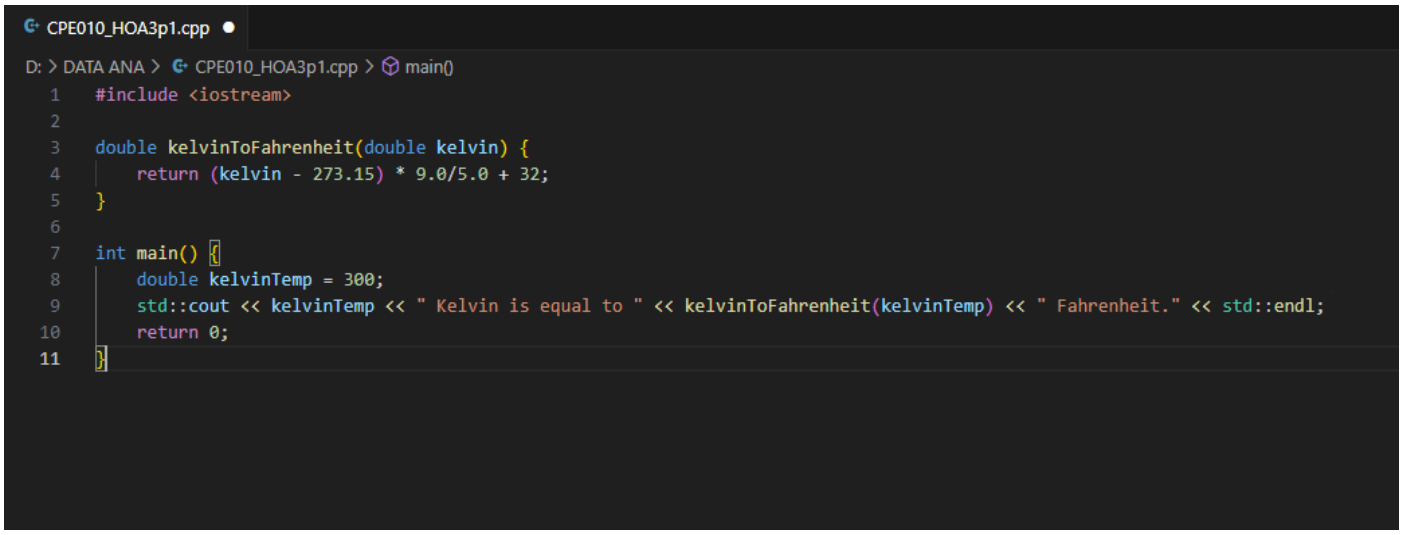
```
Output

Before swap: x = 5, y = 10
After swap: x = 10, y = 5


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

2. Kelvin to Fahrenheit Conversion

CODE:



```
CPE010_HOA3p1.cpp ●
D: > DATA ANA > CPE010_HOA3p1.cpp > main()
1  #include <iostream>
2
3  double kelvinToFahrenheit(double kelvin) {
4      return (kelvin - 273.15) * 9.0/5.0 + 32;
5  }
6
7  int main() {
8      double kelvinTemp = 300;
9      std::cout << kelvinTemp << " Kelvin is equal to " << kelvinToFahrenheit(kelvinTemp) << " Fahrenheit." << std::endl;
10     return 0;
11 }
```

OUTPUT:

```
Output

300 Kelvin is equal to 80.33 Fahrenheit.


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

3. Distance between two points

CODE:

CPE010_HOA3p1.cpp

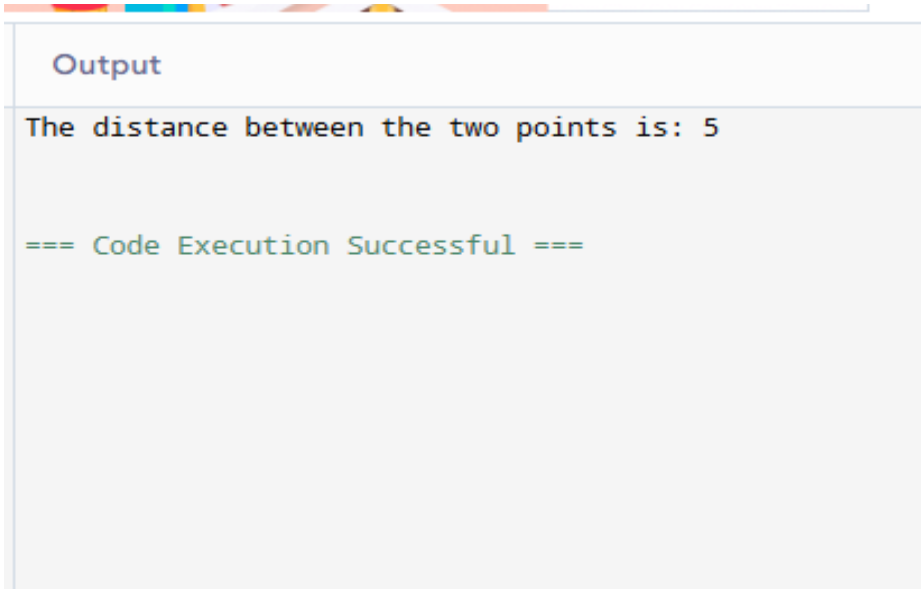
D: > DATA ANA > CPE010_HOA3p1.cpp > main()

```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 = 1, y1 = 2;
    double x2 = 4, y2 = 6;
    std::cout << "The distance between the two points is: " << distance(x1, y1, x2, y2) << std::endl;
    return 0;
}
```

OUTPUT:



4. Triangle Class

CODE:

CPE010_HOA3p1.cpp

D: > DATA ANA > CPE010_HOA3p1.cpp > Triangle > getType()

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

class Triangle {
private:
    double sideA, sideB, sideC;

public:

    Triangle(double a, double b, double c) {
        sideA = a;
        sideB = b;
        sideC = c;
    }

    double getArea() {
        double s = (sideA + sideB + sideC) / 2.0;
        return sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
    }

    double getPerimeter() {
        return sideA + sideB + sideC;
    }

    std::string getType() {
        double a = sideA, b = sideB, c = sideC;

        if (a > b && a > c) {
            double temp = c;
            c = a;
            a = temp;
        } else if (b > c) {
```

```
26
27      std::string getType() {
28          double a = sideA, b = sideB, c = sideC;
29
30          if (a > b && a > c) {
31              double temp = c;
32              c = a;
33              a = temp;
34          } else if (b > c) {
35              double temp = c;
36              c = b;
37              b = temp;
38          }
39
40          if (pow(a, 2) + pow(b, 2) > pow(c, 2)) {
41              return "Acute-angled";
42          } else if (pow(a, 2) + pow(b, 2) < pow(c, 2)) {
43              return "Obtuse-angled";
44          } else {
45              return "Right-angled";
46          }
47      }
48  };
49
50  int main() {
51      Triangle triangle(3, 4, 5);
52      std::cout << "Perimeter: " << triangle.getPerimeter() << std::endl;
53      std::cout << "Area: " << triangle.getArea() << std::endl;
54      std::cout << "Type: " << triangle.getType() << std::endl;
55
56      return 0;
57  }
```

OUTPUT:

```
Output

Perimeter: 12
Area: 6
Type: Right-angled


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

8. Conclusion

During this activity, a quick review of what we learned during our first year, in object programming, is quite an refresher for our mind to bring back what we learned, such as the swapping of two numbers and the conversation of two units, and it also highlighted the importance of a class's internal logic, as we saw with the triangle class. We revisited how to initialize an object, and how member functions can be used to perform specific tasks, like calculating area, perimeter,

and even determining a triangle's type. During this exercise it reinforced our understanding of applying these fundamental programming concepts.
9. Assessment Rubric