Activity No. 2	
Hands-on Activity 1.1 Basic C++ Programming	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** September 09, 2024
Section: CPE21S4	**Date Submitted:** September 10, 2024
Name(s): Masangkay, Frederick D.	**Instructor:** Ma’am Maria Rizette Sayo
6. Output	

Section	Answer
Header File Declaration Section	#include <iostream>
Global Declaration Section	

Class Declaration and
Method Definition Section
	class Triangle{
	    private:
	    double totalAngle, angleA, angleB, angleC;
	
	    public:
	    Triangle(double A, double B, double C);
	    void setAngles(double A, double B, double C);
	    const bool validateTriangle();
	};

Main Function	int main(){
	//driver code
	    Triangle set1(40, 30, 110);
	    if(set1.validateTriangle()){
	    cout << "The shape is a valid triangle.\n";
	    } else {
	    cout << "The shape is NOT a valid triangle.\n";
	    }
	    return 0;
	}

Method Definition	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);
	}

7. Supplementary Activity

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

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

int main() {
    int temp;
    int a = 1;
    int b = 3;

    temp = a;
    a = b;
    b = temp;
    cout << a << endl;
    cout << b;
}
```

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

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

int main () {
    int kelvin;
    float fahrenheit;

    cout << "Enter a value:";
    cin >> kelvin;

    farenheit = 1.8 * (kelvin - 273) + 32;
    cout << "Kelvin to farenheit:" <<  fahrenheit
```

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

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

int main () {
    int pointA1;
    int pointB1;
    int pointA2;
    int pointB2;
    int distance;
cout << "Enter point A1: ";
cin >> pointA1;
cout << "Enter point B1: ";
cin >> pointB1;
cout << "Enter point A2: ";
cin >> pointA2;
cout << "Enter point B2: ";
cin >> pointB2;

distance = sqrt(pow((pointA2 - pointA1), 2) + pow((pointB2 - pointB1), 2));
cout << "This is the distance between two points: " << distance;
}
```

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.’

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

class Triangle {
   private:
      double angleA, angleB, angleC, sideA, sideB, sideC, totalAngle;

   public:
      Triangle (double A, double B, double C);
      void setAngles(double A, double B, double C);
      double calculateArea();
      double calculatePerimeter();
      string triangleType();
      const bool validateTriangle();
};

Triangle::Triangle(double A, double B, double C)
{
   sideA = A;
   sideB = B;
   sideC = 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);
}

double Triangle::calculateArea()
{
    double s = (sideA + sideB + sideC) / 2;
    return sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
}

double Triangle::calculatePerimeter()
{
    return sideA + sideB + sideC;
}

string Triangle::triangleType()
{
    if (angleA > 90 || angleB > 90 || angleC > 90)
        return "Obtuse-angled";
    else if (angleA == 90 || angleB == 90 || angleC == 90)
        return "Right-angled";
    else
        return "Acute-angled";
}

int main ()
{
    Triangle set1(3, 4, 5);
    set1.setAngles(60, 60, 60);
    cout << "Area: " << set1.calculateArea() << endl;
    cout << "Perimeter: " << set1.calculatePerimeter() << endl;
    cout << "Triangle Type: " << set1.triangleType() << endl;
    return 0;
}
```

8. Conclusion

In this activity , I regained my knowledge on the basics of c++ programming language. It was insightful and challenging at the same time. For some reason, the syntax of c++ seems unknown to me but when I realized the structure of writing code on it, the pattern was obvious. I can improve my coding skill with more practice. Some native functions were not really familiar, but it brings excitement as I dig into the code. It’s not easy to re-learn a language in which I haven't touched for a long time, yet it opens some doors into new technology and application on a real world basis.

9. Assessment Rubric