using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/// <summary>

/// File name: PTangAssignment2/TriangleSolver.cs

///

/// Purpose: Create a C# console application determining the correct type of a triangle, then use Git Bash

/// for version control, NUnit for unit testing, then explain the Control Flow Graph and Cyclomatic

/// Complexity in a word document

///

/// Created by Patrick Tang

///

/// History:

/// February 14, 2017 - Created

/// - Added more code in Program.cs and TriangleSolver.cs

/// February 22, 2017 - Added comments and finished program

/// </summary>

namespace PTangAssignment2

{

public static class TriangleSolver

{

/// <summary>

/// This method analyzes the inputs from side A, side B and side C from Program.cs to determine which

/// triangle will be printed using the if statement.

/// </summary>

/// <param name="sideAInput">First input from the user (Side A)</param>

/// <param name="sideBInput">Second input from the user (Side B)</param>

/// <param name="sideCInput">Third input from the user (Side C)</param>

public static string Analyze(int sideAInput, int sideBInput, int sideCInput)

{

string triangleOutput = "";

//

// If any inputs have zeroes in it, it will pass through the next if statement. Otherwise, it will

// print the message in the else statement

//

// Note: This was needed if it happens to pass through the normal program somehow

//

if (sideAInput != 0 || sideBInput != 0 || sideCInput != 0)

{

//

// If all sides are equal, it will print an "Equilateral Triangle" message

//

// Example: 5, 5, 5

//

if (sideAInput == sideBInput && sideAInput == sideCInput)

{

triangleOutput = "This is an Equilateral triangle";

}

//

// If two sides are equal and the total of the first two sides are less than the third side, it

// will print an "Isosceles Triangle" message

//

// Example: 2, 2, 3

//

else if ((sideAInput == sideCInput && (sideAInput + sideCInput) > sideBInput) ||

(sideBInput == sideCInput && (sideBInput + sideCInput) > sideAInput) ||

(sideAInput == sideBInput && (sideAInput + sideBInput) > sideCInput))

{

triangleOutput = "This is an Isosceles triangle";

}

//

// This area makes sure that all sides are specifically put in correctly to make a Scalene

// Triangle. It checks the first section to see if all sides are less than the other. In

// the same input, it checks if the total of the first two sides are less than the third

// side. If it passes these tests, it will print a "Scalene Triangle" messaage

//

// Example: Side B < Side C < Side A and (Side B + Side C) < Side A

//

else if (((sideAInput < sideBInput && sideBInput < sideCInput) && ((sideAInput + sideBInput) > sideCInput)) ||

((sideAInput < sideCInput && sideCInput < sideBInput) && ((sideAInput + sideCInput) > sideBInput)) ||

((sideBInput < sideAInput && sideAInput < sideCInput) && ((sideBInput + sideAInput) > sideCInput)) ||

((sideBInput < sideCInput && sideCInput < sideAInput) && ((sideBInput + sideCInput) > sideAInput)) ||

((sideCInput < sideBInput && sideBInput < sideAInput) && ((sideCInput + sideBInput) > sideAInput)) ||

((sideCInput < sideAInput && sideAInput < sideBInput) && ((sideCInput + sideAInput) > sideBInput)))

{

triangleOutput = "This is a Scalene triangle";

}

//

// If any of the inputs doesn't fail after the Scalene test, an invalid triangle message

// will be printed

//

// Example: 1, 1, 3 = Instant Fail

//

else

{

triangleOutput = "This does not form any valid triangles";

}

}

else

{

triangleOutput = "No triangles can be formed with any zeroes";

}

return triangleOutput;

}

}

}