12.0-Understanding delegates

Program

```
using System;

namespace Test
{
    internal class Program
    {
        public delegate double MathOperation(double x, double y);
        public static void Main(string[] args)
        {
            Console.Write("Enter the first number: ");
            double number1 = double.Parse(Console.ReadLine());

            Console.Write("Enter the second number: ");
            double number2 = double.Parse(Console.ReadLine());

            MathOperation powerDelegate = Power;
            MathOperation rootDelegate = Root;

            double resultPower = PerformOperation(number1, number2, powerDelegate);
            double resultRoot = PerformOperation(number1, number2, rootDelegate);

            // Display results
            Console.WriteLine($"Power Result: {resultPower}");
            Console.WriteLine($"Root Result: {resultRoot}");
```

```
        }

        static double Power(double x, double y)
        {
            return Math.Pow(x, y);
        }

        static double Root(double x, double y)
        {
            if (y != 0)
                return Math.Pow(x, 1.0 / y);
            else
            {
                Console.WriteLine("Cannot calculate root for zero!");
                return double.NaN; // Not a Number
            }
        }

        static double PerformOperation(double x, double y, MathOperation operation)
        {
            return operation(x, y);
        }
    }
}
```

Exercise

Write a C# program that demonstrate the usage of delegates in C# to perform mathematical

operations such as power and root calculations.

The program prompts the user to input two numbers: number1 and number2.

The user provides the values for number1 and number2 via the console.

The program defines a delegate MathOperation, which represents a method that takes two double parameters and returns a double.

Two delegate instances are created: powerDelegate and rootDelegate.

Two static methods, Power and Root, are defined to perform power and root calculations, respectively.

The Power method calculates the result of raising number1 to the power of number2.

The Root method calculates the result of taking the yth root of number1.

A static method PerformOperation is defined to perform mathematical operations using the provided delegate.

This method takes number1, number2, and a MathOperation delegate as parameters.

It invokes the delegate with the provided parameters and returns the result.

The program calculates the result of power and root operations using the input numbers and the respective delegate instances.

It displays the results of both operations (resultPower and resultRoot) on the console.

Hint

Use powerDelegate and rootDelegate are instances of the MathOperation delegate.

They are initialized with references to the Power and Root methods, respectively.

The Power method calculates the result of raising x to the power of y using Math.Pow.

The Root method calculates the result of taking the yth root of x.

Ensure you understand how Math.Pow and the exponentiation operation (1.0 / y) work.

Explanation

This program uses of delegates to perform mathematical operations such as exponentiation and

root calculation. Upon execution, the program prompts the user to enter two numbers, which are then utilized to perform the mathematical operations. Here's a detailed explanation:

The MathOperation delegate is declared to represent a method that takes two double parameters (x and y) and returns a double result. This delegate is used to reference methods that perform specific mathematical operations.

In the Main method, the program prompts the user to enter the first and second numbers via the console. These inputs are then parsed into double variables, number1 and number2.

Two delegate instances, powerDelegate and rootDelegate, are created and initialized with references to the Power and Root methods, respectively. These methods correspond to the Power and Root mathematical operations.

The Power method calculates the result of raising x to the power of y using Math.Pow. If y is zero, indicating an attempt to calculate the root of zero, the method displays an error message and returns double.NaN (Not a Number) to signify an invalid result.

Similarly, the Root method calculates the yth root of x using Math.Pow(x, 1.0 / y). If y is zero, it displays an error message and returns double.NaN.

The PerformOperation method takes x, y, and a MathOperation delegate as parameters. It invokes the delegate with the provided parameters (x and y) and returns the result of the mathematical operation.

Finally, after performing the operations, the program displays the results of the power and root calculations using string interpolation ($"") to format the output.