**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **SingletonPatternExample**.
2. **Define a Singleton Class:**
   * Create a class named Logger that has a private static instance of itself.
   * Ensure the constructor of Logger is private.
   * Provide a public static method to get the instance of the Logger class.
3. **Implement the Singleton Pattern:**
   * Write code to ensure that the Logger class follows the Singleton design pattern.
4. **Test the Singleton Implementation:**
   * Create a test class to verify that only one instance of Logger is created and used across the application.

**Program.cs**

**using System;**

**using System.Security.Cryptography.X509Certificates;**

**public class Logger**

**{**

**private static Logger \_instance;**

**private Logger()**

**{**

**Console.WriteLine("Logger instance is created");**

**}**

**public static Logger GetInstance()**

**{**

**if(\_instance == null)**

**{**

**\_instance = new Logger();**

**}**

**return \_instance;**

**}**

**public void Log(string message)**

**{**

**Console.WriteLine("Log: " + message);**

**}**

**}**

**class Program**

**{**

**static void Main()**

**{**

**Logger logger1 = Logger.GetInstance();**

**logger1.Log("This is the first instance");**

**Logger logger2 = Logger.GetInstance();**

**logger2.Log("This is the second instance");**

**if (Object.ReferenceEquals(logger1, logger2)) {**

**Console.WriteLine("Only one instance exists");**

**}**

**else**

**{**

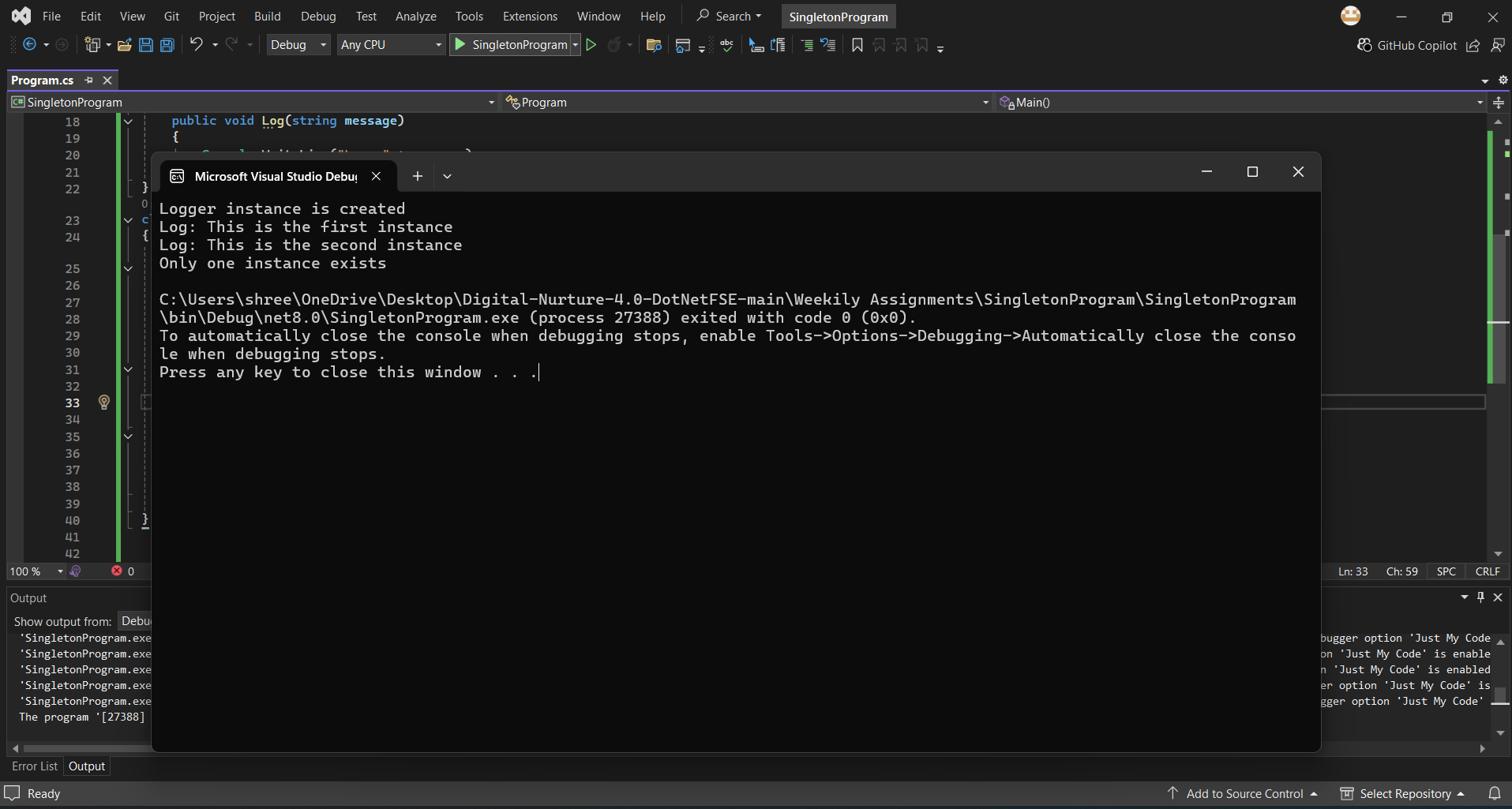
**Console.WriteLine("Multiple instance exist");**

**}**

**}**

**}**

**OUTPUT:**

****