**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.

# THE CODE:

1. using System;
2. public class Logger
3. {
4. // Step 2.1: Private static instance of the same class
5. private static Logger? \_instance;
6. // Step 2.2: Private constructor to prevent external instantiation
7. private Logger()
8. {
9. Console.WriteLine("Logger instance created.");
10. }
11. public static Logger? Instance { get => Instance1; set => Instance1 = value; }
12. public static Logger? Instance1 { get => \_instance; set => \_instance = value; }
13. // Step 2.3: Public static method to return the single instance
14. public static Logger GetInstance()
15. {
16. if (Instance == null)
17. {
18. Instance = new Logger();
19. }
20. return Instance;
21. }
22. // A sample method to simulate logging
23. public void Log(string message)
24. {
25. Console.WriteLine($"[LOG]: {message}");
26. }
27. }
28. class Program
29. {
30. static void Main(string[] args)
31. {
32. Console.WriteLine("Testing Singleton Logger...");
33. // Retrieve two logger instances
34. Logger logger1 = Logger.GetInstance();
35. Logger logger2 = Logger.GetInstance();
36. // Log messages using both references
37. logger1.Log("This is the first message.");
38. logger2.Log("This is the second message.");
39. // Test if both instances point to the same object
40. if (ReferenceEquals(logger1, logger2))
41. {
42. Console.WriteLine("Both logger instances are the same (Singleton confirmed).");
43. }
44. else
45. {
46. Console.WriteLine("Logger instances are different (Singleton failed).");
47. }
48. }
49. }

# THE RESULT:

