**WEEKLY TASK - 4**

**NAME : SANJEEV N**
**DATE : 07-08-2025**

**PROBLEM 1**

**Create two threads to read two separate text files**

```
class Program
{
   static void Main()
   {
      Thread thread1 = new Thread(() =>
ReadFile("C:\\Users\\sanjeev.n\\Desktop\\qwert\\file1.txt"));
      Thread thread2 = new Thread(() =>
ReadFile("C:\\Users\\sanjeev.n\\Desktop\\qwert\\file2.txt"));

      thread1.Start();
      thread2.Start();

      thread1.Join();
      thread2.Join();

      Console.WriteLine("Both threads have completed.");
   }

   static void ReadFile(string filePath)
   {
      try
      {
         string content = File.ReadAllText(filePath);
         Console.WriteLine($"Contents of {filePath}:\n{content}\n");
      }
      catch (Exception ex)
      {
         Console.WriteLine($"Error reading {filePath}: {ex.Message}");
      }
   }
}
```

```
Contents of C:\Users\sanjeev.n\Desktop\qwert\file1.txt:
Hello world!


Contents of C:\Users\sanjeev.n\Desktop\qwert\file2.txt:
Hey Folks!!!!!!!

Both threads have completed.
```

**Same two files, try to read using the Task Async await.**

```
class Program
{
    static async Task Main()
    {
        Task<string> task1 = ReadFileAsync("C:\\Users\\sanjeev.n\\Desktop\\qwert\\file1.txt");
        Task<string> task2 = ReadFileAsync("C:\\Users\\sanjeev.n\\Desktop\\qwert\\file2.txt");

        string[] results = await Task.WhenAll(task1, task2);

        Console.WriteLine("Contents of file1.txt:\n" + results[0]);
        Console.WriteLine("Contents of file2.txt:\n" + results[1]);
    }

    static async Task<string> ReadFileAsync(string filePath)
    {
        try
        {
            string content = await File.ReadAllTextAsync(filePath);
            return content;
        }
        catch (Exception ex)
        {
            return $"Error reading {filePath}: {ex.Message}";
        }
    }
}
```

```
Contents of file1.txt:
Hello world!

Contents of file2.txt:
Hey Folks!!!!!!!
```

**PROBLEM 2**

**Create delegate use case between teacher class and student class:**
**- teacher method should have test_completed() method passed as delegate to student**
**- student class should have a method write_test() which will in turn call the parent delegate.**

```
namespace school
{
   using System;

   namespace DelegateDemo
   {
      public delegate void TestDelegate();

     public class Teacher
      {
         public void TestCompleted()
         {
            Console.WriteLine("Teacher: Test has been evaluated.");
         }
      }

      public class Student
      {
         public void WriteTest(TestDelegate testCallback)
         {
            Console.WriteLine("Student: Writing the test...");
            Console.WriteLine("Student: Finished writing the test.");

            testCallback();
         }
      }
      class Program
      {
         static void Main(string[] args)
         {
            Teacher teacher = new Teacher();
            Student student = new Student();

            student.WriteTest(teacher.TestCompleted);

            Console.WriteLine("Process complete.");
```

```
        }
    }
  }

}
```

```
Student: Writing the test...
Student: Finished writing the test.
Teacher: Test has been evaluated.
Process complete.
```