LAB-8

2100030723

Q1)

```
using System;

public delegate void ArrayHandler<T>(object sender, ArrayEventArgs<T> e);

public class ArrayEventArgs<T> : EventArgs
{
   public int Id { get; }
   public T Value { get; }
   public string Message { get; }

   public ArrayEventArgs(int id, T value, string message)
   {
      Id = id;
      Value = value;
      Message = message;
   }
}

public class CustomArray<T>
{
   private T[] array;
   private int startIndex;

   public event ArrayHandler<T> OnChangeElement;
   public event ArrayHandler<T> OnChangeEqualElement;
```

```
  public CustomArray(int length, int startIndex)
  {
    if (length <= 0)
      throw new ArgumentException("Length must be greater than zero.");

    this.array = new T[length];
    this.startIndex = startIndex;
  }

  public int FirstIndex => startIndex;
  public int LastIndex => startIndex + array.Length - 1;
  public int Length => array.Length;

  public T this[int index]
  {
    get
    {
      CheckIndex(index);
      return array[index - startIndex];
    }
    set
    {
      CheckIndex(index);
      T oldValue = array[index - startIndex];
      array[index - startIndex] = value;

      if (!oldValue.Equals(value))
      {
        OnChangeElement?.Invoke(this, new ArrayEventArgs<T>(index, value, "Element value
changed."));
      }
```

```
        if (value.Equals(index))
        {
            OnChangeEqualElement?.Invoke(this, new ArrayEventArgs<T>(index, value, "Element value equals index."));
        }
      }
    }

    private void CheckIndex(int index)
    {
      if (index < startIndex || index >= startIndex + array.Length)
        throw new IndexOutOfRangeException($"Index {index} is out of range.");
    }
}

public class Program
{
    public static void Main(string[] args)
    {
      // Example usage of CustomArray
      CustomArray<int> intArray = new CustomArray<int>(5, 0);

      // Subscribe to events
      intArray.OnChangeElement += IntArray_OnChangeElement;
      intArray.OnChangeEqualElement += IntArray_OnChangeEqualElement;

      // Changing elements
      intArray[0] = 1;
      intArray[1] = 2;
      intArray[2] = 3;
```

```
        intArray[3] = 4;
        intArray[4] = 5;
    }

    private static void IntArray_OnChangeElement(object sender, ArrayEventArgs<int> e)
    {
        Console.WriteLine($"Element at index {e.Id} changed to {e.Value}. Message: {e.Message}");
    }

    private static void IntArray_OnChangeEqualElement(object sender, ArrayEventArgs<int> e)
    {
        Console.WriteLine($"Element value equals index at index {e.Id}. Message: {e.Message}");
    }
}
```

OUTPUT :