Lab8: Delegate and Events

Name: Gondrala Mani Sai

Id number: 2100031545

Task **Delegates and events** refers to task **Collections**

To add the following new functionalities to the project created in task Collections:

Include two events in the **CustomArray** type:

- The **OnChangeElement** event occurs when the indexer changes the element value (if the old and new element values match, the event is not raised)
- The **OnChangeEqualElement** event occurs if a value equal to the index of the changed element is written to the element (if the old and new values of the element match, the event is not raised)

Use the **ArrayHandler** delegate to create the event.

The event handler takes two parameters: an object **sender** - a reference to the **CustomArray** instance that generated the event, and an event argument **ArrayEventArgs <T> e**. In the event argument, write in the **Id** field the index by which the user changes the element of the **CustomArray** array, in the **Value** field - the new value of the element by the Id index, in the **Message** field - an arbitrary string message.

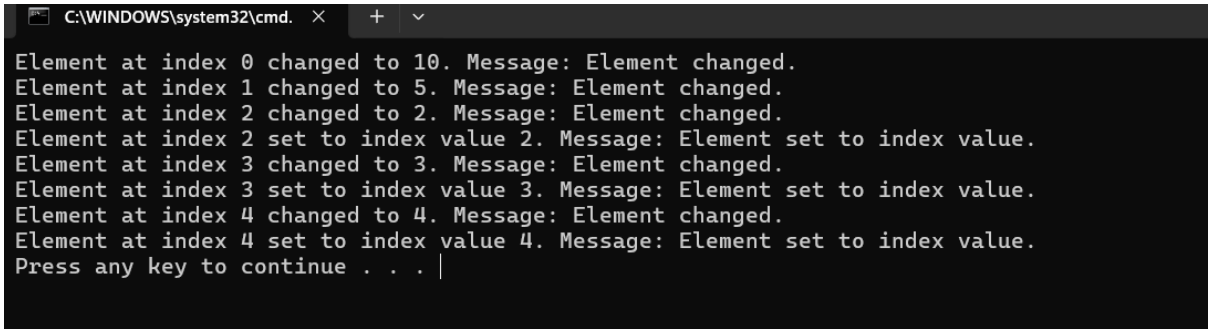
Solution:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab_8
{
    delegate void ArrayHandler<T>(object sender, ArrayEventArgs<T> e);
    internal class CustomArray<T>
    {
        private T[] array;
        private int startIndex;

        // Events
        public event ArrayHandler<T> OnChangeElement;
        public event ArrayHandler<T> OnChangeEqualElement;

        // Constructor to create an empty user array
        public CustomArray(int startIndex, int length)
        {
            if (length <= 0)
                throw new ArgumentException("Length must be greater than zero.");

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

        // Indexer with event raising
        public T this[int index]
        {
```

```
            get
            {
                ValidateIndex(index);
                return array[index - startIndex];
            }
            set
            {
                ValidateIndex(index);

                // Raise OnChangeElement event if the new value is different from
the current value
                T oldValue = array[index - startIndex];
                if (!oldValue.Equals(value))
                {
                    array[index - startIndex] = value;
                    OnChangeElement?.Invoke(this, new ArrayEventArgs<T>(index,
value, "Element changed."));
                }

                // Raise OnChangeEqualElement event if the new value is equal to
the index
                if (index.Equals(value))
                {
                    OnChangeEqualElement?.Invoke(this, new
ArrayEventArgs<T>(index, value, "Element set to index value."));
                }
            }
        }

        // Method to validate the index
        private void ValidateIndex(int index)
        {
            if (index < startIndex || index >= startIndex + array.Length)
                throw new IndexOutOfRangeException($"Index '{index}' is out of
range.");
        }
    }
    }


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab_8
{
    internal class ArrayEventArgs<T> : EventArgs
    {
        public int Id { get; set; }       // Index of the changed element
        public T Value { get; set; }      // New value of the element
        public string Message { get; set; }   // Optional message

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

}

Output:


C:\WINDOWS\system32\cmd.
Element at index 0 changed to 10. Message: Element changed.
Element at index 1 changed to 5. Message: Element changed.
Element at index 2 changed to 2. Message: Element changed.
Element at index 2 set to index value 2. Message: Element set to index value.
Element at index 3 changed to 3. Message: Element changed.
Element at index 3 set to index value 3. Message: Element set to index value.
Element at index 4 changed to 4. Message: Element changed.
Element at index 4 set to index value 4. Message: Element set to index value.
Press any key to continue . . .