2100030320

M. Bhavana

EPAM – LAB-8

IN-LAB:

TASK1: Task **Delegates and events** refers to task **Collections**

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

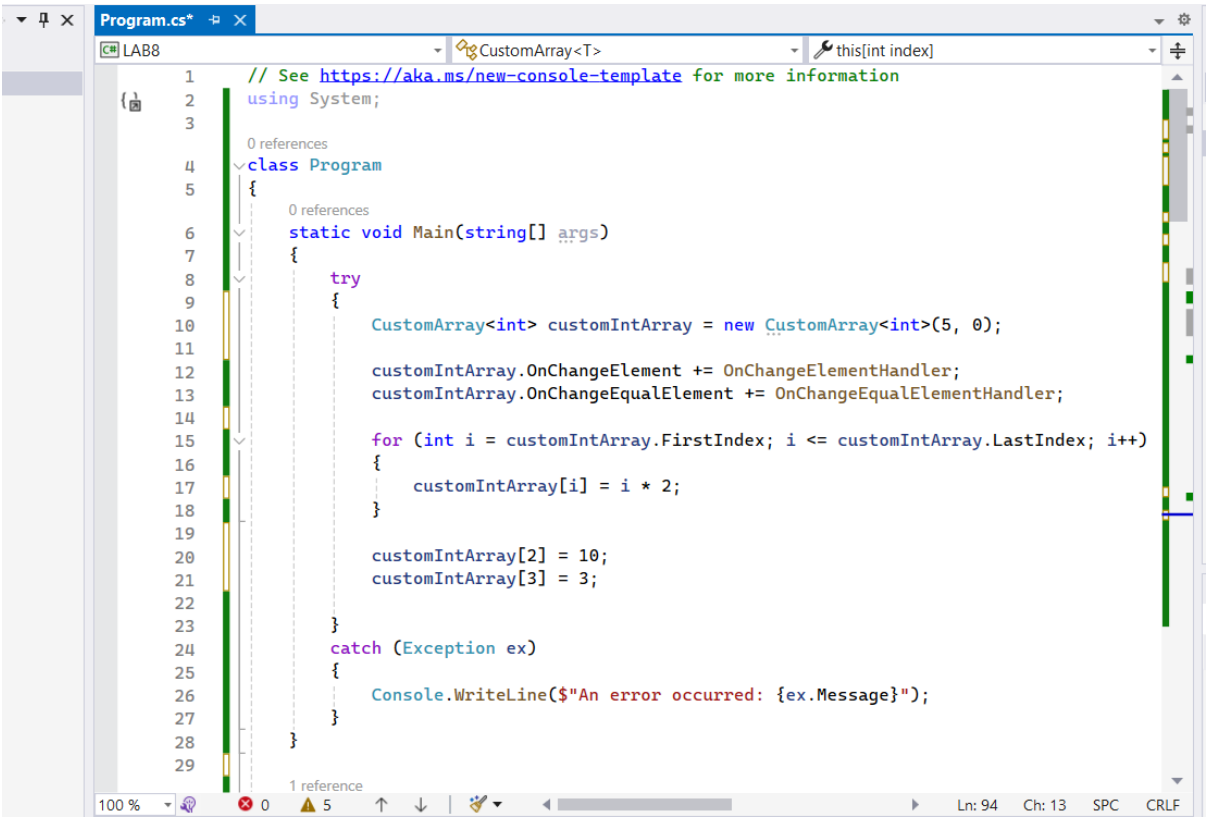
Include two events in the **CustomArray** type:

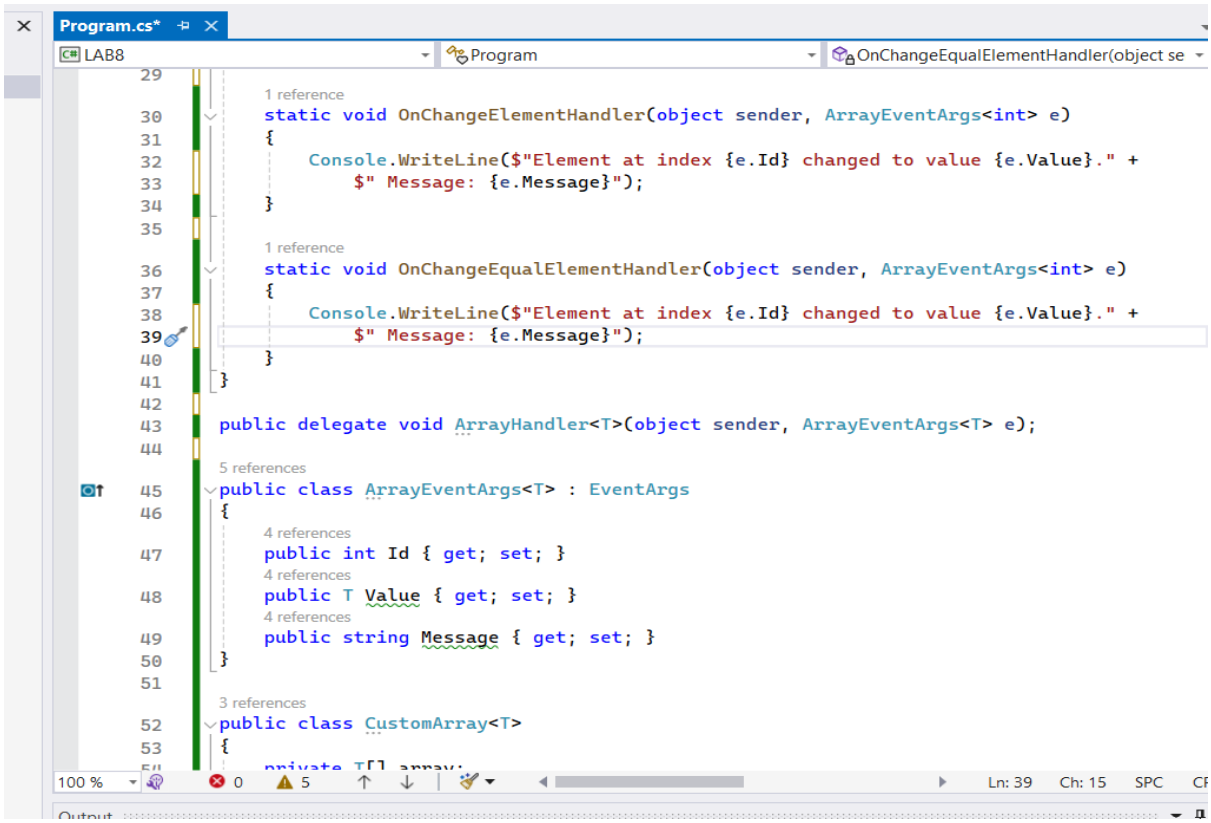
- The **OnChangeElement** event occurs when the indexer changes the element value (if the old and new element values match, the event is not raised)
- The **OnChangeEqualElement** event occurs if a value equal to the index of the changed element is written to the element (if the old and new values of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

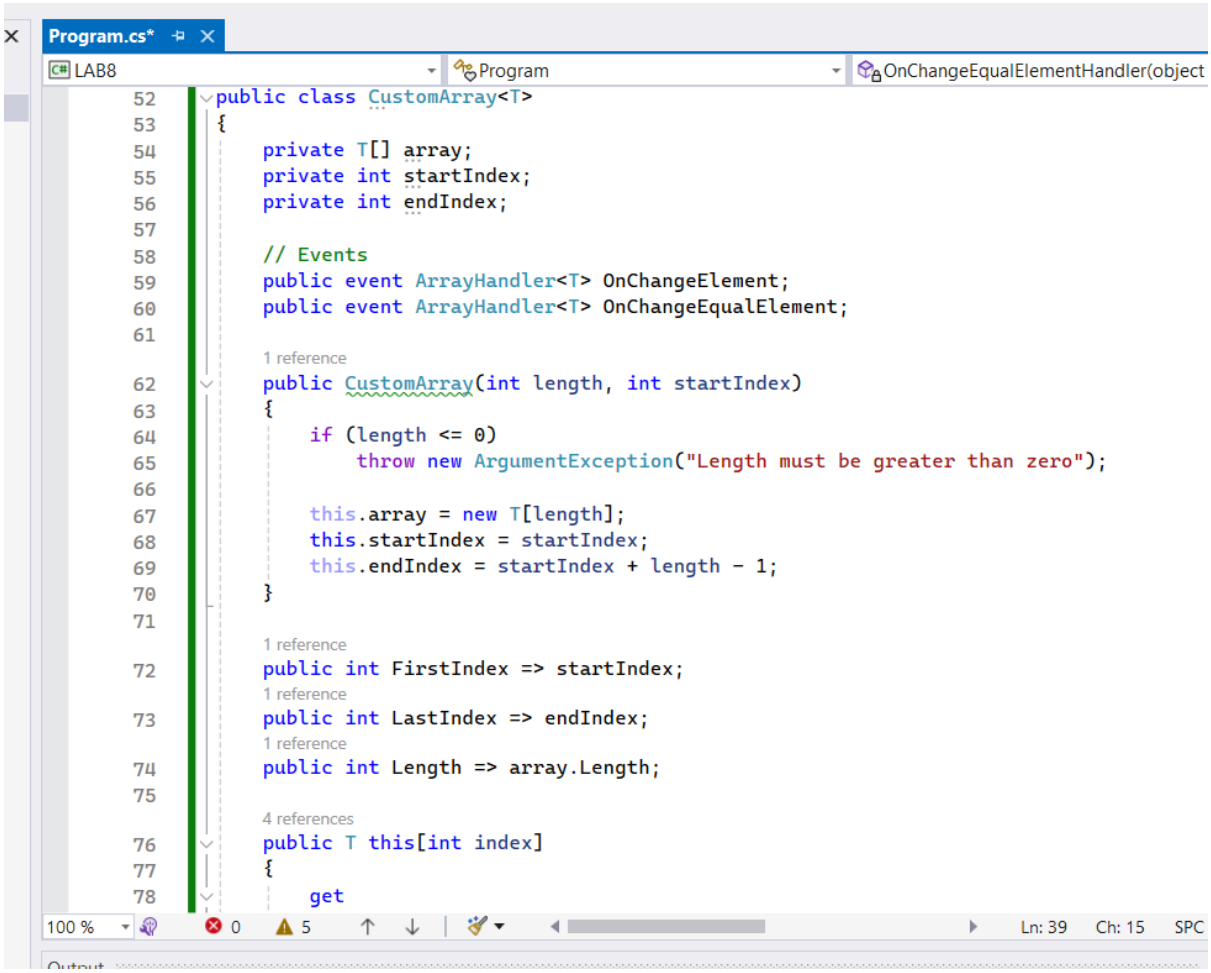
Use the **ArrayHandler** delegate to create the event.

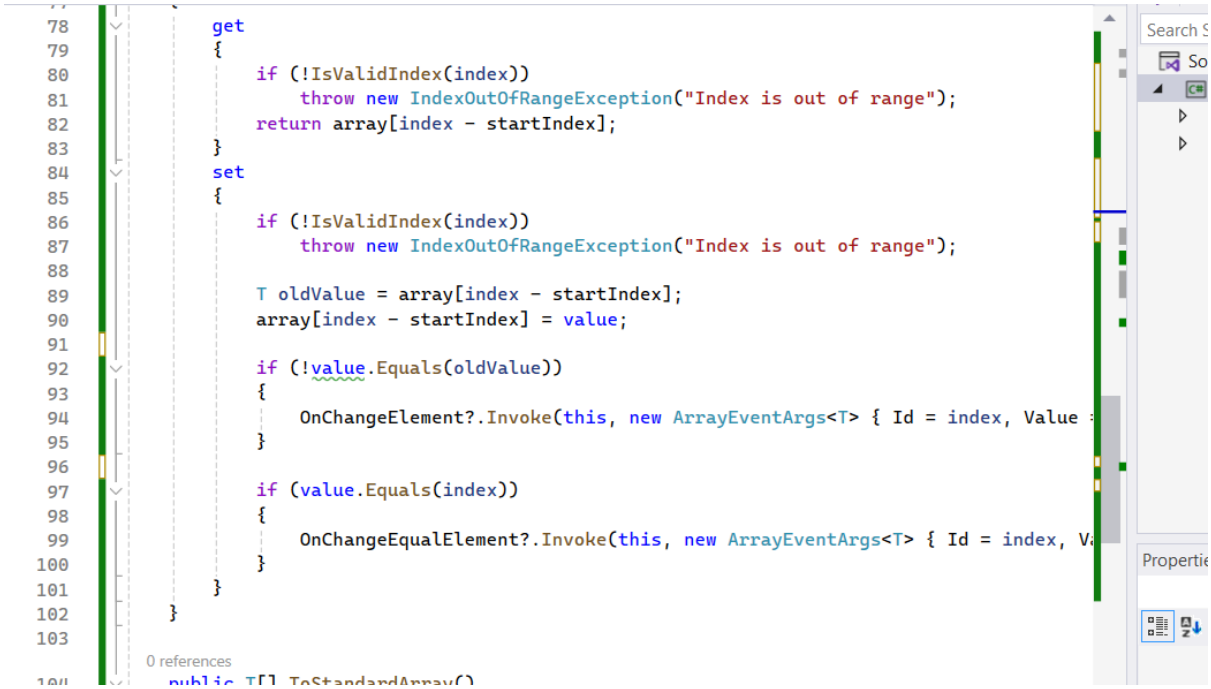
The event handler takes two parameters: an object **sender** - a reference to the **CustomArray** instance that generated the event, and an event argument **ArrayEventArgs <T> e**. In the event argument, write in the **Id** field the index by which the user changes the element of the **CustomArray**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

Code:

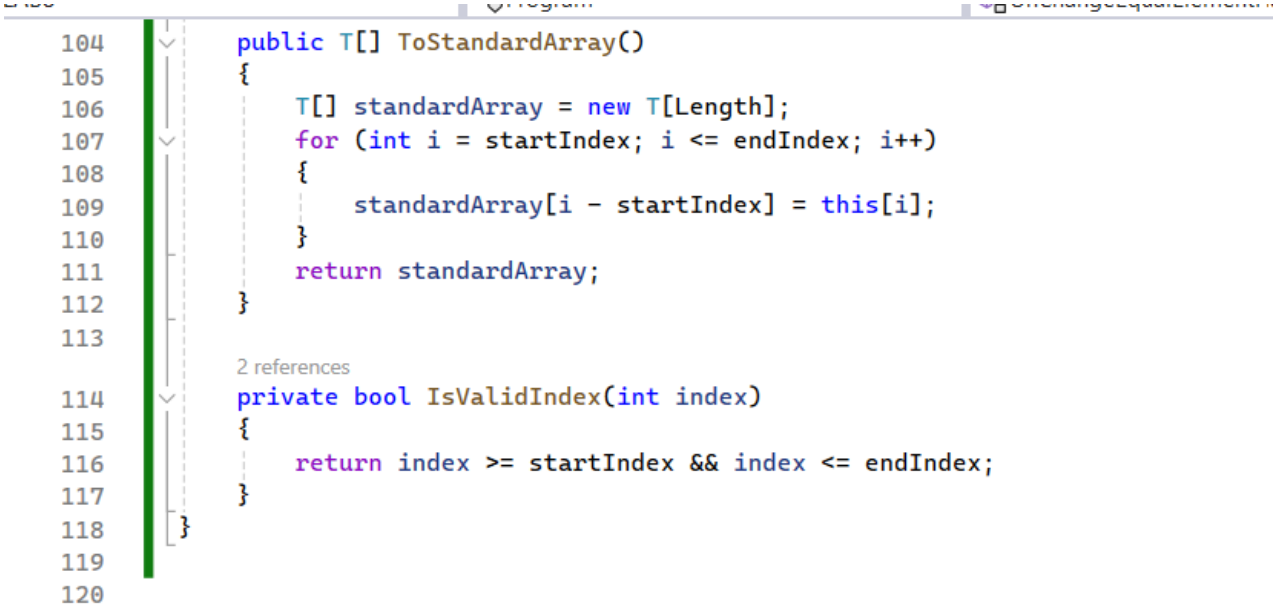



Program.cs*
LAB8
Program
OnChangeEqualElementHandler(object se
1 reference
static void OnChangeElementHandler(object sender, ArrayEventArgs<int> e)
{
Console.WriteLine($"Element at index {e.Id} changed to value {e.Value}." +
$" Message: {e.Message}");
}
1 reference
static void OnChangeEqualElementHandler(object sender, ArrayEventArgs<int> e)
{
Console.WriteLine($"Element at index {e.Id} changed to value {e.Value}." +
$" Message: {e.Message}");
}
}
public delegate void ArrayHandler<T>(object sender, ArrayEventArgs<T> e);
5 references
public class ArrayEventArgs<T> : EventArgs
{
4 references
public int Id { get; set; }
4 references
public T Value { get; set; }
4 references
public string Message { get; set; }
}
3 references
public class CustomArray<T>
{
100 %
0
5
Ln: 39
Ch: 15
SPC


Program.cs*
LAB8
Program
OnChangeEqualElementHandler(object
public class CustomArray<T>
{
private T[] array;
private int startIndex;
private int endIndex;
// Events
public event ArrayHandler<T> OnChangeElement;
public event ArrayHandler<T> OnChangeEqualElement;
1 reference
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;
this.endIndex = startIndex + length - 1;
}
1 reference
public int FirstIndex => startIndex;
1 reference
public int LastIndex => endIndex;
1 reference
public int Length => array.Length;
4 references
public T this[int index]
{
get
100 %
0
5
Ln: 39
Ch: 15
SPC

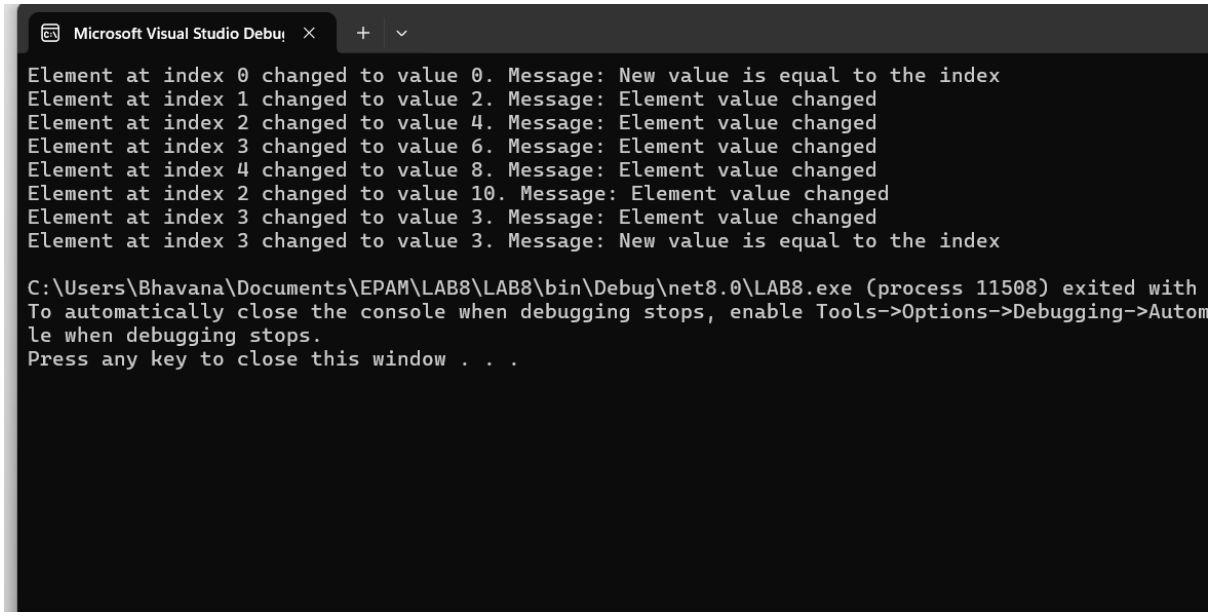


```
78          get
79          {
80              if (!IsValidIndex(index))
81                  throw new IndexOutOfRangeException("Index is out of range");
82              return array[index - startIndex];
83          }
84          set
85          {
86              if (!IsValidIndex(index))
87                  throw new IndexOutOfRangeException("Index is out of range");
88
89              T oldValue = array[index - startIndex];
90              array[index - startIndex] = value;
91
92              if (!value.Equals(oldValue))
93              {
94                  OnChangeElement?.Invoke(this, new ArrayEventArgs<T> { Id = index, Value =
95              }
96
97              if (value.Equals(index))
98              {
99                  OnChangeEqualElement?.Invoke(this, new ArrayEventArgs<T> { Id = index, V
100             }
101         }
102     }
103
        0 references
104     public T[] ToStandardArray()
```



```
104     public T[] ToStandardArray()
105     {
106         T[] standardArray = new T[Length];
107         for (int i = startIndex; i <= endIndex; i++)
108         {
109             standardArray[i - startIndex] = this[i];
110         }
111         return standardArray;
112     }
113
        2 references
114     private bool IsValidIndex(int index)
115     {
116         return index >= startIndex && index <= endIndex;
117     }
118 }
119
120
```

Output:


Microsoft Visual Studio Debu
Element at index 0 changed to value 0. Message: New value is equal to the index
Element at index 1 changed to value 2. Message: Element value changed
Element at index 2 changed to value 4. Message: Element value changed
Element at index 3 changed to value 6. Message: Element value changed
Element at index 4 changed to value 8. Message: Element value changed
Element at index 2 changed to value 10. Message: Element value changed
Element at index 3 changed to value 3. Message: Element value changed
Element at index 3 changed to value 3. Message: New value is equal to the index
C:\Users\Bhavana\Documents\EPAM\LAB8\LAB8\bin\Debug\net8.0\LAB8.exe (process 11508) exited with
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Autom
le when debugging stops.
Press any key to close this window . . .