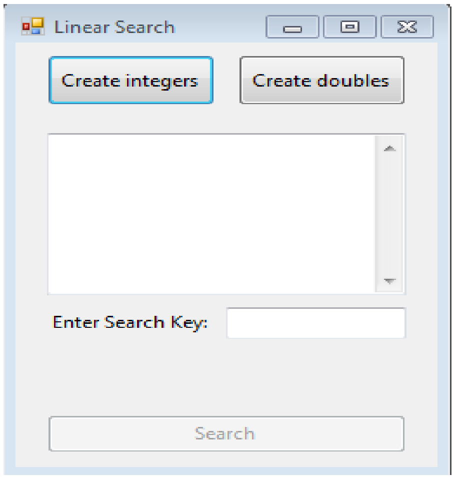
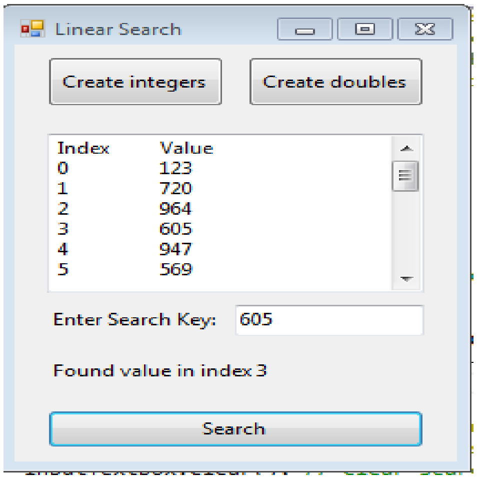
This assignment has 2 programming questions.

Programming Question 1: Generic Linear Search

Specification:

Create a generic method, Search, which implements the linear search algorithm. This method should compare the search key with each element in the list until the search key is found or until the end of the list is reached. If the search key is found, return its location in the list; otherwise, return -1.





```
Index     Value
0         123
1         720
2         964
3         605
4         947
5         569
```

Enter Search Key: 605

Found value in index 3

Create a Windows Forms application with the above UI that inputs and searches a list of integers and a list of doubles. Provide buttons that the user can click to randomly generate int and double values. (For double type numbers, you can use the random number generated * 2.5 to store the double values.) Display the generated values in a TextBox, so the user knows what values they can search for.

Create a class and include search method as follows

```
 public static int Search<T>(T key, List<T> aList)
          where T : IComparable<T>  { ... }   //this allows you to use CompareTo method
```

Programming Question 2: Customer Maintenance Application with delegate and event

Specification:

This customer maintenance application is partially developed. The project is located in this topic. You download this application and finish it up.

Add a CustomerList class

1. Add a class named CustomerList to the project, and add the following members to the class

Properties	Description
Count	An integer that indicates how many Customer objects are in the list.

[index]	An indexer that provides access to the Customer at the specified position. Code is provided for this property
	`public Customer this[int i]`
	`        {`
	`            get`
	`            {`
	`                return customers[i];`
	`            }`

```
            set
            {
                customers[i] = value;
                Changed(this);
            }
        }
```

Methods	Description
Add(customer)	Adds the specified Customer object to the list.
Remove(customer)	Removes the specified Customer object from the list.
Fill()	Fills the list with customer data from a file using the GetCustomers() method of the CustomerDB class.
Save()	Saves the customers to a file using the SaveCustomers() method of the CustomerDB class.

2. Modify the Customer Maintenance form (frmCustomers.cs) to use this class. To do this, you will need to use the Fill and Save methods of the CustomerList object instead of methods of the CustomerDB class. In addition, you will need to use a for loop instead of a foreach loop when you fill the list box.
3. 3. Run the applications to test it.

Add a delegate and an event to the CustomerList Class

4. Add a delegate named ChangeHandler to the CustomerList class. This delegate should specify a method with a void return type and a CustomerList parameter.
5. Add an event named Changed to the CustomerList class. This event should use the ChangeHandler delegate and should be raised any time the customer list changes.
6. Modify the Customer Maintenance form (frmCustomers.cs) to use the Changed event to save the customers and refresh the list box any time the list changes. To do that you will need to code an event handler that has the signature specified by the delegate. You will need to wire the event to the event handler, and you will need to remove any unnecessary code from the event handlers for the Save and Delete buttons.
7. Run and test and application.

When you complete these two projects, zip them and submit them in Eagle online for grading.