Peer Evaluation for Lab 2 – Chapters 13, 14 and 15

Your name: (Your lab is the one being evaluated)	Robert Moore
Name(s) of peer evaluator(s)	" "
Date:	4/23/2018

<u>Instructions</u>

You should have already completed Lab 2. After you and a peer have evaluated your work, you will submit this evaluation along with screen shots and source code indicated in moodle. You may make corrections to your work as a result of the evaluation.

In Class Exercises – 15-1 #3	
Completed Exercise?	Yes
• Created a class diagram in visual studio? Screen shot included?	Yes
• Customer class implements the ICloneable interface?	Yes
o ICloneable is added to the class heading?	Yes
o Clone method is implemented with the signature expected by the framework?	Yes
• Test calls Clone to make copies of the customer object.	Yes
• Screen shot of test running is included?	Yes
• Source code includes Customer class as well as test for Clone?	yes
In Class Exercises – 15-2 #2	
Completed Exercise?	Yes
• Created a class diagram in visual studio? Screen shot included?	yes
• CustomerList class implements the IEnumerable interface?	Yes
o IEnumerable is added to the class heading?	yes
o GetEnumerator method is implemented with the signature expected by the framework?	Yes
	Yes
• Test fills a CustomerList and iterates through the list using a foreach loop?	Yes
• Screen shot of test running is included?	Yes
• Source code includes CustomerList class as well as test for the foreach loop?	Yes

In Class Exercises – 13-1	

Completed Exercise?	
• Created a class diagram in visual studio? Screen shot included? (**YES**) • CustomerList class includes delegate and event? (**YES**) ◦ Delegate ChangeHander is declared? (**YES**) ◦ Event Changed is declared?(**YES**) ◦ All methods that change the list "broadcast" the Changed event?(**YES**) • User interface responds to the Changed event?(**YES**) ◦ Method is written to handle the changed event, the signature of which matches the delegate?(**YES**) ◦ Form load event handler "wires up" the method to the event?(**YES**) • Screen shot of application running is included? (**YES**) • Source code includes CustomerList class as well as event handlers/methods from UI? (**YES**)	

In Class Exercises – AlmostEmpty, Deck and Blackjack	
Completed Exercise? • Created a class diagram in visual studio? Screen shot included? • Deck class includes delegate and event? ◦ Delegate EmptyHander is declared? ◦ Event AlmostEmpty is declared? ◦ Empty HandleEmpty method is created? ◦ Constructor "wires up" the AlmostEmpty event and the empty HandleEmpty method? ◦ Deal method "broadcasts" the AlmostEmpty event? • User interface responds to the AlmostEmpty event? ◦ Method is written to handle the AlmostEmpty event, the signature of which matches the delegate? ◦ Form load event handler "wires up" the method to the event? • Screen shot of application running is included? • Source code includes Deck class as well as event handlers/methods from UI that are related to handling the event?	

Programming style for all programs	
Is proper indentation used? Is each property/method indented properly? Is each control structure indented properly?	
Are comments used appropriately?	
Do variable names use camel case? (camelCase for example)	
Do property/method names use Title Case (or Pascal Case?)	

General comments and notes:

Screen Shots and Source Code

Top window — CloneCustomer (Running) - Microsoft Visual Studio

Class Diagram showing:
- IEnumerator<Customer>
- ICloneable
- CustomerList (Class): Fields: currentCustomer, customers; Properties: Count, Current, IEnumerator.Cu..., this; Methods: Add, CustomerList, Dispose, getCustomers, HandleChanged
- Customer (Class): Fields: email, firstName, lastName; Properties: Email, FirstName, LastName; Methods: Clone, Customer (+ 1..., GetDisplayText, ToString
- Program (Static Class)

Clone Customer dialog:
John Mendez, jmendez@msysco.com
Copies: 4   Clone
John,Mendez,jmendez@msysco.com
John,Mendez,jmendez@msysco.com
John,Mendez,jmendez@msysco.com
John,Mendez,jmendez@msysco.com
Exit

Error List — Entire Solution — 0 Errors — 0 Warnings — 0 of 5 Messages

Bottom window — Class1.cs code:

```csharp
        }
        [Test]
        public void AdjustCustomer()
        {
            this.customer.FirstName = "Rob";
            this.customer.LastName = "Moore";
            this.customer.Email = "moorerb@my.lanecc.edu";

            Assert.AreEqual("Rob", this.customer.FirstName);
            Assert.AreEqual("Moore", this.customer.LastName);
            Assert.AreEqual("moorerb@my.lanecc.edu", this.customer.Email);
        }
        [Test]
        public void CreateClones()
        {
            // Run the previous test
            this.AdjustCustomer();

            // Clone the Customer
            int timesToDo = 1000;
            do {
                customerList.Add((Customer)customer.Clone());
                timesToDo--;
            } while (timesToDo > 0);

            Assert.AreEqual(1000, customerList.Count);

        }
```

File Edit View Project Build Debug Team Tools Test Analyze Window Help   Full Screen   Quick Launch (Ctrl+Q)   moorerb   M

ClassDiagram1.cd   Form1.cs* X   Form1.cs [Design]   CustomerList.cs   Class1.cs   NuGet: CloneCustomerTests   Program.cs   Customer.cs   X
CloneCustomer   CloneCustomer.Form1   HandleChanged(CustomerList customers)

```csharp
17              InitializeComponent();
18              Changed = new ChangeHandler(HandleChanged);
19          }
20
21          private Customer customer;
22          // Valid for 15-1 - List <Customer> customers = new List<Customer>();
23          // Updated for version 15-2
24          // private List<Customer> customers;
25
26          private CustomerList customers = new CustomerList();
27          // Part of the 13-1 portion of the assignment
28          public delegate void ChangeHandler(CustomerList customers);
29
30          public event ChangeHandler Changed;
31
32          public void HandleChanged(CustomerList customers)
33          {
34
35              lstCustomers.DataSource = customers.getCustomers();
36
37          }
38          private void Form1_Load(object sender, System.EventArgs e)
39          {
40              customer = new Customer("John", "Mendez", "jmendez@msysco.com");
41              lblCustomer.Text = customer.GetDisplayText();
42          }
43
44
45          private void btnExit_Click(object sender, System.EventArgs e)
```

119 %

Ready   Ln 33   Col 10   Ch 10   INS   ↑ 2   99*   CS234   master

File Edit View Project Build Debug Team Tools Test Analyze Window Help   Full Screen   Quick Launch (Ctrl+Q)   moorerb   M

ClassDiagram1.cd   Form1.cs   Form1.cs [Design]   CustomerList.cs* X   Class1.cs   NuGet: CloneCustomerTests   Program.cs   Customer.cs   X
CloneCustomer   CloneCustomer.CustomerList   Current

```csharp
13          {
14              // Part of Event Handler
15              Changed = new ChangeHandler(HandleChanged);
16              // Starts at 0
17              this.currentCustomer = 0;
18          }
19          private List<Customer> customers = new List<Customer>();
20          /// <summary>
21          /// Return a Count of the number of entres in customers
22          /// </summary>
23          public int Count => customers.Count;
24          /// <summary>
25          /// Grab the current Customer at the Index Point
26          /// </summary>
27          public Customer Current => this.customers[this.currentCustomer];
28          /// <summary>
29          /// Grab the current Customer at the Index Point
30          /// </summary>
31          object IEnumerator.Current => this.customers[this.currentCustomer];
32          public Customer this[int i] => customers[i];
33          public delegate void ChangeHandler(CustomerList customerList);
34
35          public event ChangeHandler Changed;
36
37          public void HandleChanged(CustomerList customerList) {
38              // Is not supposed to do anything - its just so one event is bound already - so we dont have to
39              // check if we're already bound, or deal with other things that may be bound. - In the future tho that
40              // would probably be better etiquette.
41          }
```

119 %

Ready   Ln 27   Col 73   Ch 73   INS   ↑ 2   99*   CS234   master