**Week 3:** [**Abstract Classes and Interfaces**](https://classes.lanecc.edu/course/view.php?id=75355&section=3)

|  |
| --- |
| **Things to do Weeks 3 and 4** |

* **Reading Quiz 3**
  + Read chapters 13, (page 418 - 423), 14 (page 470 - 473) and 15 in the C# text.
  + Complete Reading Quiz 3.
* **Lab 2**
  + Complete exercise 15-1 #3 and 15-2 #2  on pages 505 and 506 of the text using your version of the Customer and CustomerList class from the end of 233N.  Test the method Clone in the Customer class and the foreach loop in the CustomerList class in a console app or an nUnit unit test.
  + Complete exercise 13-1 parts 11 - 14 on page 437 and 438 of the text.  Use the version of the Customer and CustomerList classes as well as the UI that is provided in the starting files for this lab.
  + Design and implement the Hand and PrivateTrain classes following the specification discussed in class.  Use inheritance, interfaces, events and delegates as appropriate.  Test both of the classes using nUnit unit tests.
  + Participate in the forum - Lab 2 Questions as necessary.
  + Submit lab 2.
* [**Lab 3**](https://classes.lanecc.edu/mod/assign/view.php?id=1563437)
  + Complete a Windows Forms application that allows a user to play one or more hands of Blackjack against the computer (as the dealer).
  + Participate in the forum - Lab 3 Questions as necessary.
  + Submit lab 3.

## Lab 2

The objective of this lab is to familiarize you with creating implementing Interfaces in C#.  This information is contained in chapter 15 of your text.

Complete the programming problems described below.

* Complete exercise 15 - 1 #3 as described on page 505 of the text.  Add the interface and the method to your Customer class from [lab 1](https://classes.lanecc.edu/mod/assign/view.php?id=1563424).  Test the interface and method in a console app or an nUnit unit test.
* Complete exercise 15 - 2 #2 as described on page 506 of the text.  Add the interface and method(s) to your CustomerList class (not CustomerList2) from the end of 233N.  Test the interface and method(s) in a console app or an nUnit unit test.
* Complete exercise 13 - 1 parts 11 - 14 on page 437 and 438 of the text.  Use the starting files provided rather than your previous versions of the Customer Maintenance application.
* Design, implement and test the PrivateTrain and Hand classes for Mexican Train Dominos following the specification discussed in class.  Write a set of unit tests for each class.  Implement the Hand and PrivateTrain classes in C#, one method at a time, testing as you complete each method.  Use inheritance, abstract classes and interfaces as appropriate in your implementation.

A maximum of 20 points will be awarded for the lab.

In class section students should:

* Complete the implementation, test and debug the class in Visual Studio.  Add a class diagram
* Download the peer evaluation form for lab 2.  Complete the peer evaluation with a classmate using the form as a guide.  Include with the peer evaluation document:
  + screen shots illustrating that all tests run successfully
  + the NEW source code for the classes and test programs that you wrote in each problem
  + class diagram for each class you wrote of changed
* Upload the peer evaluation document you created in moodle.

## Lab 3

The objective of this lab is to allow you to create an Object Oriented Windows Forms application that is slightly more complex in C#.

Complete the a GUI version of BlackJack that allows at least one user to compete against the Dealer.  Use the classes you created and tested in earlier labs as well as a user interface you design.

A maximum of 20 points will be awarded for the lab.

In class section students should:

* Create an Action Control Event (ACE) chart for the application.
* Create a list of form level variables the application will use.
* Create a list of UI related methods and methods that tie your classes and the UI together.  Feel free to "steal and modify" any methods from Concentration that may be helpful.
* Write an algorithm in pseudocode that describes the logic involved in any method or event handler that is logically complex
* Complete the implementation, test and debug the application in Visual Studio.
* Download the peer evaluation form for lab 3.  Complete the peer evaluation with a classmate using the form as a guide.  Include with the peer evaluation document:
  + screen shots illustrating the functionality of the applicatioin in problem
  + the source code for the methods, event handlers and variable declarations that you wrote to solve the problem.  It is NOT necessary to include source code for your classes UNLESS you created new classes that were not part of earlier labs.
* Upload the peer evaluation document you created in moodle.