# Exercise 1

The program we worked on in class can be found in the **Classroom** **Files** in the D2L **content** area. Modify the program found there such that it will have the functionality below. DO NOT use functions from .NET or the System.Array class to implement these methods; do them on your own using only C# statements (for/while loops, assignment statements, etc). The methods can be static just as we did in class.

* Add a method that will insert a new entry into the array at a given index -- it should not overwrite the existing entry; think through this carefully.
* Add a method that will find the index where a string should be inserted into a sorted array. For example, for an array containing the following values,

{April, June, May}

* + Alice would return zero, since it is before April
  + John would return one, since it is before June but after April
  + Paula would return three, since it is after May

Note that this function is very similar to the binary search that we implemented in class. The only difference is that instead of returning -1, we return the insertion point. You'll want to test this carefully, particularly for 'edge conditions'.

* Add a method that will delete the entry at a given index. Make certain you don't leave a hole.
* Now that you have created these methods, create a program that will do the following:

Read the names to be sorted from a text file containing 12 names rather than coding them in the program itself. (Create this file yourself using a text editor.) You can assume there will be never be more than 16 names in the list.

Present the user with a menu similar to the following:

1. Add
2. Delete
3. Display All
4. Quit

If the user selects **Add** or **Delete**, prompt for the name to be found or added. Use the new methods to add or delete the given name. If the name is not found for delete, display an appropriate message.

If the user selects **Display All**, print the names in the array to the console on a single line after each command.

Redisplay the menu until the user decides to quit. You do not have to validate the user input.

Insert the name, print the list, prompt for another name.

* Answer the following questions in a text or Word document:

1. If we call the number of elements in the array 'N', how many elements need to be moved if we insert after the last element? before the first element?
2. If we call the number of elements in the array 'N', how many elements need to be moved if we delete the last element? the first element?
3. If we get strings at random, what will be the average number of elements we need to move (in terms of N) to add or delete?
4. Which would be faster, to insert the new name as we did above, or to add the name at the bottom of the list and resort the list? Why? Will the other choice ever be faster?
5. **Bonus** **question**: a search of a sorted array of length N will take at most how many comparisons before the entry (or insertion point) is found?