**CSc 110 Lab 9**

**Review**

From last lab, an example of a very simple Battleship game is here: [Battleship.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/code/Battleship.java)

**Objectives:**

* Creating an object class, Contact.
* Declaring an new instance of a contact object.
* Using the methods.
* Understanding toString.
* Using PrintStream to write to a file. Remember the [FileIO notes](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/14_FileIO.pdf)

***Lab Evaluations***

[**https://evals.csc.uvic.ca/IstanuEval/**](https://evals.csc.uvic.ca/IstanuEval/)

If you experience any difficulties, please contact:  Kathryn Wilson, ECS 460, itsupport@csc.uvic.ca, local 5900

***Exercise 1*** Demo program (define, initialize, search.)

Download [Contact.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/code/Contact.java).  Compile it.  Open it for examination.

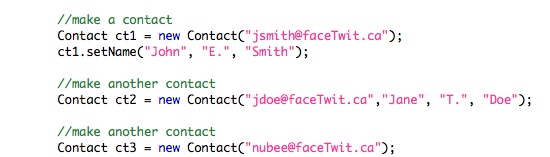
The TA will identify the following defining characteristics of a class specification for you as you tour the code.

* Attributes and methods.
* The idea of private and public access.
* The constructor(s).
  + No return type (void or otherwise).
  + Has the same name as the class.
  + Is invoked when you use the 'new' keyword.
* Usually methods in an object specification are instance methods and are therefore the static keyword is not applicable.

***Exercise 2***

Download [TestContact.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/code/TestContact.java) into the same folder as Contact.java  Examine it, compile it, run it.

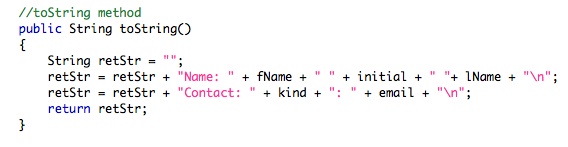
Add the following code to the main.



**We will add some more tests for the other methods (board and projector examples).**

***Exercise 3***

Enhancing the Contact class:  
  
The toString method:  Every object class in java has a toString method that allows us to print the contents of an object with a single command rather than using the access methods to get information, one element at a time.  The toString method is invoked automatically anytime the + operator is used in a string context or a print function is called.  Put the following code into the Contact class:



Save and compile.  
  
Go back into main in TestContact and print each contact that you have created so far.  You can now print all the data for the Contact ct1 by merely using the command: System.out.println(ct1);

***Exercise 4***

Add a new attribute, a private String variable called alias.  Assign it's default value to the empty String.  
  
Make a new constructor that takes two String parameters, the first the email and the second the alias.  This will allow you to store a contact by alias.  
Add alias to the output of the toString method too.  
  
Assure that the Contact class compiles.  
  
Declare a new contact in your TestContact main and initialize it with an email and an alias to test the new constructor, altered toString and the attribute.   Be sure to  print the new contact.  
  
**Discussion:  other methods that could be added to Contact: pros, cons and possible implementations.**

***Exercise 5***

Demo program:   
Download [ContactList.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/code/ContactList.java) into the same folder as Contact.java.  Download and use [TestContactList.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab9/code/TestContactList.java) instead of TestContact.java.  
  
A ContactList object contains an array of Contacts.  We are making only one ContactList (in the TestContactList class) although you should know that we could make multiple contact lists.   
  
Don't let the wrappers and the more complex syntax frighten you.  If one has an array 'myArray' of Strings, for example, then myArray[0] is one of those Strings.  We know that String variables have a method associated with them called equals(anotherString) as in:  
  
String s1 = "hello";  
String s2 = "goodbye";  
String s3 = "hello";  
So the command: s1.equals(s2) will return false while  s1.equals(s3) will return true.  
  
The reference myArray[0].equals(s2) is just using the first element in the array as a name for the String at that position in the array, and comparing that String to s2.  
  
You should be able to read and understand ContactList.java.  You should be comfortable with instance methods and variables as opposed to static methods and variables.

**a) Complete the method save() in ContactList.java, test it.**  
  
**b) Consider writing a new method in ContactList to findByAlias: what would need to be done to write and test such a method? Where would these changes be made?**

***Exercise 6* - Sorting the Contacts By Last Name**

Use Bubble Sort or the sort that used a helper method to find a minimum, which is a kind of Selection Sort (remember from Lab 7, reviewed in Lab 8?) to sort the contacts in your contact list, by last name.

First you will need to add a method to Contact.java, which gets the last name only.

Then you will work in ContactList.java to write a method "sort()", which may call on additional methods.

Use the String method "compareTo". Find the java documentation by searching "java String" and looking at the API.

Test your sort method: cList.sort(); by printing or saving to file before and after. Ensure that it is working.

***Example of a different Object and its use***

See this object [Monster.java](https://connex.csc.uvic.ca/access/content/group/766cb70a-9ef1-463a-9e71-78eb524c1e1d/lab%20resources/Lab9/code/Monster.java)the code which uses it: [Game.java](https://connex.csc.uvic.ca/access/content/group/766cb70a-9ef1-463a-9e71-78eb524c1e1d/lab%20resources/Lab9/code/Game.java)and an input file for Game.java: [enemies.txt](https://connex.csc.uvic.ca/access/content/group/766cb70a-9ef1-463a-9e71-78eb524c1e1d/lab%20resources/Lab9/code/enemies.txt)

These must all be in the same directory in order for Game.java to compile and run. It should output a file called outfile.txt.

***Looking back...***

In this lab you'll have practiced and/or seen examples of:

* Java Object classes: attributes (public and private), instance methods, constructors, toString...
* Using and testing instances of Java objects: creating new instances, invoking non-static methods, printing objects...
* Objects inside arrays
* The 'this' keyword
* Writing to a file with PrintStream

The End.