Course: CSC 115 (J)

Lab 02: Object Oriented Programming

Objectives

- Basic Input Methods
- Understand basic concepts related to object oriented programming.
  - Class definitions.
  - Instance variables.
  - Access modifiers, getters/setters.
  - The toString method.
  - A unit tester in the form of the main method.
- Make progress on Assignment 1.

Part I – Basic Input Methods

1. Keyboard Input

The Java Scanner class, part of the *java.util* package, simplifies the commonly used task of getting input from the computer's keyboard. For example, here is code can be used to get name from the user and display a Hello message to the user.

```
Scanner input = new Scanner (System.in);
System.out.print("What is your name? ");
String name = input.next();
System.out.println("Hello " + name + "..!);
```

Visit http://docs.oracle.com/javase/1.5.0/docs/api/java/util/Scanner.html to see all the available methods / functions the scanner class provide. Feel free to experiment with methods that you have never used.

2. Command Line Arguments

A java program can be passed with any number of arguments from the command line, i.e., if required, we can pass arguments to the main method. These parameters allow the user to specify configuration information when the application is launched. Syntax to specify these parameters is:

```
java MyProgram arguments
```

The *arguments* are separated by spaces or tabs. If a space is needed in an argement then that argument is enclosed by double-quotes. Secondly, all arguments are received as a

single String array. Individual parameters can be formatted back from String to their corresponding data types.

- Write a program that accepts a single command line argument that outputs a word or phrase.
- Write a program that accepts a single command line argument (a number) and prints the numbers 1 through the input number.
- Write a program that accepts a name of a person and outputs a sentence with that person's name included in it.

Further details on command line arguments can be found at
http://docs.oracle.com/javase/tutorial/essential/environment/cmdLineArgs.html

3. File Input

The Scanner class can also be used for breaking down formatted input into tokens and translating individual tokens according to their data type. A file input can be specified instead of using *System.in* (keyboard) for reading tokens as under:

```
Scanner s = new Scanner(new BufferedReader(new FileReader("inFile.txt")));
```

Functions *s.next(), s.nextInt(),* and *s.nextDouble()* etc can be used to read tokens in specific formats.

Part II – Basic concepts of Object Oriented Programming

Designing a Class

- Participate in a group discussion designing a class of objects with data and methods associated with it.
- Design and implement your own class. Make sure to test it using a unit tester.

Exercise 01: Building the Contact Class

For this exercise, you will implement a ContactList class that keeps track of a list of Contacts. Each Contact object will store a name, a phone number, and an email address. Your ContactList will allow a user to add, remove, or print the contacts in the list.

- Create a Contact class that will store information about a Contact, including:
  - Name
  - Phone Number
  - Email Address

- Write the following methods in the Contact class:
    - Getters for the class fields
    - String toString()
        - returns the information in the Contact as a single String
- Write a main method in the Contact class that creates a Contact object and tests all the above methods.

Exercise 02: Building the ContactList class

- Create a ContactList class that stores an array of Contact objects.
- Write the following methods:
    1. void add(String name, String phone, String email)
        - adds a new Contact object to the end of the array.
    2. String toString()
        - returns the ContactList as a printable String including all Contacts in the list.
    3. void remove(String name)
        - removes the first Contact from the list that matches the specified name.
        - use the **equals** method when comparing two Strings to see if they contain the same word.
        - Don't worry about preserving the order of the items in the list - this means that an easy way to remove an item in the array is to overwrite it with the last item in the array, and then decrease the size of the list by 1.
- Write a main method in the ContactList class that creates a ContactList object and tests all the above methods. It is best to test each method as you finish them before moving on to the next, so you have a better idea where any emerging errors have come from.

Part III – Assignment 1

- Work on Assignment 1, feel free to ask your lab TA for help with any problems you encounter.