COE528 (Fall 2016)
Lab2

General Lab Rules
All the necessary files of this lab should be in lab2 directory.
All the java files in this lab should have the following package declaration:

```
package coe528.lab2;
```

Duration: one week.

Objectives

- Use Java Collections framework. Implement interface.
- Implement and specify procedures with requires, modifies and effects clauses.

Exercise 1: Using ArrayList

In this lab, we are going to learn how to use few classes of the Java Collections framework.

A helpful tutorial site on Java Collections can be found at:

http://docs.oracle.com/javase/tutorial/collections/index.html

You might find the following websites helpful for doing this exercise:

http://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html

http://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html

In the Netbeans program, click on Project > New Project and save it as "Ex1" on your lab2 directory.

1(a) Create a new class called “Book”:

- A `Book` should include three pieces of information as instance variables – title (type *String*), isbn number (type *String*) and author (type *String*).

- The `Book` class should have a constructor that takes three arguments (title, isbn number and author) and initializes the three instance variables.

- Provide getter methods for each instance variable.

- Implement the `toString` method of this class so that the method returns the title.

1(b) Create a new class called “BookList” that contains a list of books. Following is an incomplete declaration for the `BookList` class. Implement:

- the body of the constructor,
- `addBook` method and
- the `main` method

as indicated by the comments.

```
import java.util.*;

public class BookList {

   private ArrayList<Book> books;

   public BookList() {

    //Write the code for the constructor:

    //Initialize the instance variable books




   }

   public void addBook(String t, String i, String a) {

    //Write the code for addBook

    //Creates a book with its title t, isbn i and author a. Then adds the newly

    //created book to the arraylist books.




   }

   public String toString() {

     return books.toString();

   }
   public static void main(String[] args) {

       BookList bList = new BookList();

      //use the method addBook to add the following books

      //to the list bList in the order as shown below.

      //title: “Calculus”, isbn: “1234”, author: “Goldstein”

      //title: “Java”, isbn: “5678”, author: “Gosling”

      //title: “Algorithms”, isbn: “4629”, author: “Cormen”

      System.out.println(bList);

   }

}
```

1(c) After completing the above exercises 1(a) and 1(b), run the `main` method of `BookList` class. You will find that the titles of the books are printed in alphabetically unsorted manner as follows:

[Calculus, Java, Algorithms]

This is because with an ArrayList, the elements are kept in the order in which they were inserted into the list.

Our next goal is to sort the list of books (based on their titles) and then print the sorted list. For this purpose, we will use `Collections.sort(List<T> list)` method.

- Add the following `sort()` method in the `BookList` class:

```
public void sort() {

  Collections.sort( books );

}
```

- Add the following two lines at the end of `main` method of the `BookList` class:

```
bList.sort();

System.out.println(bList);
```

When you will try to compile the updated `BookList` class, it won't compile. To fix this compilation problem, the `Book` class must implement the `Comparable<Book>` interface. You need to do the following changes in the `Book` class:

- Modify the class header of the `Book` class so that now it implements the `Comparable<Book>` interface.

- Provide the implementation of the following method in the `Book` class:

```
/**
* Returns a negative integer, zero, or a positive integer depending on the title of
* this book being lexicographically less than, equal to, or greater than the title
* of the specified book b. [Hint: Use compareTo method of String class]
*/
public int compareTo(Book b)
```

Now when you run the `main` method of the `BookList` class again, you should see the following output:

[Calculus, Java, Algorithms]

[Algorithms, Calculus, Java]

Exercise 2: Procedural Abstractions

To help you complete this assignment, you may find the following useful:

The specification for a procedural abstraction contains:

(a) Requires: this clause defines the conditions under which the procedure will work. This is an optional clause.

(b) Modifies: this clause identifies all modified inputs. This is an optional clause.

(c) Effects: this clause defines the behavior of the procedure.

Make sure you respect specifications. This means that a method without Requires clause should accept all inputs (total implementation), and that a method with Requires clause should verify the clauses if appropriate.

In the Netbeans program, click on Project > New Project and save it as "Ex2" on your lab2 directory.

Create a new class called `Palindrome`. In this class:

- **Specify** and **implement** a procedure called `isPalindrome` that determines whether or not a string is a palindrome. (A palindrome reads the same backward and forward; an example is “deed’.) This method should be public and static, and take a single string parameter. The method should return true if the specified string is a palindrome and false otherwise.

- **Specify** and **implement** the `main` method of the class `Palindrome`. The `main` method tests if each of the *command-line arguments* is a palindrome or not by calling the method `isPalindrome`, and then prints either "palindrome" or "Not a palindrome" for that argument.

Submitting your lab

You must submit your lab electronically at least 24 hours prior to the start of your scheduled lab period for Lab 3.

You must include the duly filled and signed standard cover page with your submission. The cover page can be found on the departmental web site: Standard Assignment/Lab Cover Page

If you did the lab on a Departmental computer, you can do the following:

```
cd coe528
zip -r lab2.zip lab2
submit coe528 lab2 lab2.zip
```

If you did the lab on your own computer, zip the lab2 folder (remember to do this recursively so that all sub-folders are included), then transfer the zip file to a Departmental machine, logon to a Departmental machine which can be done remotely) and type in the submit command:

```
submit coe528 lab2 lab2.zip
```