Computer Science 271 – Fall 2017
Lab #3

Lab: Thursday, 9/21/17 in HS 101C
Due: Wednesday, 9/27/17 at 8:00 PM in D2L

Objective: ADTs using Arrays

The goal of this lab is to get you familiar with implementing an ADT and estimating the big-O runtime for its methods.

1. Provide a Java class called `ArrayOrderedSet` that implements the `OrderedSet` interface provided on D2L. This interface is used to represent an ordered set of `Comparable` objects stored in *ascending order*. (If you are not familiar with the `Comparable` interface, please review it from the Java API.) Your implementation class must use a generic array to store the elements.

   The specific behavior expected from each method is described below:

   - `boolean insert(T element)` - inserts `element` in set, if it is not already present, in ascending order. `True` if insertion successful, `False` otherwise.
   - `boolean contains(T element)` - `True` if `element` exists in set; `False` otherwise.
   - `boolean remove(T element)` - removes `element` if it is present in set. `True` if deletion successful, `False` otherwise.
   - `T get(int index)` – retrieves `element` at position `index;` returns `null` if `index` is invalid.
   - `int size()` - current size of set
   - `OrderedSet<T> first(int k)` - returns the first `k` (or less, if set size is less than `k`) elements of the ordered set as a new `OrderedSet`
   - `OrderedSet<T> last(int k)` - returns the last `k` (or less, if set size is less than `k`) elements of the ordered set as a new `OrderedSet`
   - `OrderedSet<T> union(OrderedSet<T> set)` - returns the union (elements in either set) of this ordered set and the parameter ordered set. A new `OrderedSet` is created and returned.
   - `OrderedSet<T> inter(OrderedSet<T> set)` - returns the intersection (elements in both sets) of this ordered set and the parameter ordered set. A new `OrderedSet` is created and returned.
   - `OrderedSet<T> diff(OrderedSet<T> set)` - returns the difference (elements in this set that are not in the parameter set) between this ordered set and the parameter ordered set. A new `OrderedSet` is created and returned.

2. Note the following:
   a. For the last five methods above, a *shallow copy* of the `T` objects into the returned `OrderedSet` is sufficient.

b. The class declaration line of your class should be

```
public class ArrayOrderedSet<T extends Comparable<T>>
                                    implements OrderedSet<T>
```

c. When creating or resizing your internal array, note that you have to do something like the following (don’t worry about warnings):

```
array = (T[])(new Comparable[size_of_your_array]);
```

3. Provide a standard `toString()` implementation in your class.

4. Just above the header for each of the public methods in your class, in comments, write the big-O runtime for that method. For example, `//O(n)`.

5. Include the file `TestOrderedSet.java`, from D2L, in your NetBeans project, and set it to be the main file. Run this file to make sure your implementation is correct.

Submission

To complete this lab, submit to D2L a single, zipped up folder, `Lab3.zip`, containing your entire NetBeans project folder. The submission dropbox folder for this lab is **Lab 3**.