BİL-211 Object Oriented Programming and Design
Assignment - 1
Assigned: June,2 2014
Due: June,17 2014 11:59PM
Submission: Compress all the files (.rar, .zip etc.) and send them to myucesan@etu.edu.tr by email.
Rules: Late submissions are not allowed. Plagiarism is strictly forbidden, all that take part will be punished according to the regulations of the university.

In this assignment, you are going to implement a data structure called SortedElements with the following functionality. A SortedElements is a data structure that keeps a list of objects. But these objects should always be kept sorted, either in increasing or decreasing order. The items below are the methods that your SortedElements class should provide:

- SortedElements(E[] N) : Constructs a SortedElements object with the objects in N. N can either be sorted or not. The objects should be sorted in increasing order by default. Note that the same object can appear more than once the list.
- get(int rank) : Returns the object at the given rank in the object.
- insert(E n) : Adds n to the implicit SortedElements object (to the correct index).
- remove(int index) : Removes the element at the specified index.
- length() : returns the length of the implicit object.
- unite(SortedElements A, SortedElements B) : Combines two given SortedElements object in *increasing* order (whatever the original orders of A and B are). Returns this as a new object.
- reverse(SortedElements A) : Reverses the numbers in A and returns it as a new object in reversed order.
- setify(SortedElements A) : Returns a new SortedElements object in the same order with A, by removing duplicate numbers.

You can use any sorting algorithm (for instance : `http://en.wikipedia.org/wiki/Selection_sort`) and any Java component that we discussed in class. As you may have already noticed, your class should be generic. You can assume the elements in the object implements the Comparable interface in Java. The class should be immutable. (How should the methods like remove be, to be immutable? Remember String class)

You should also write a JUnit class that tests your class. Try to test your class in every way as you can think of, so that the class works perfect. But you don't have to think of the cases where the given array is very large or the elements in the array do not implement Comparable interface. Write your code using clear Javadoc comments. Use and follow piazza for any questions or updates.

At the end, submit the following compressed items:

1. Your SortedElements class.
2. Your JUnit class.
3. Your HTML file produced by using javadoc.