COMP 2011 Data Structures

Assignment One

Firm Deadline: 23:59, 12 October 2014

Total Points: 100 (20 points for each question)

Programming projects on Page 76 in the textbook (Robert Lafore, Data Structures & Algorithms in Java, 2nd Edition, Sams, 2002).

Programming Projects

Writing programs to solve the Programming Projects helps to solidify your understanding of the material and demonstrates how the chapter's concepts are applied.

2.1 To the `HighArray` class in the `highArray.java` program (Listing 2.3), add a method called `getMax()` that returns the value of the highest key in the array, or –1 if the array is empty. Add some code in `main()` to exercise this method. You can assume all the keys are positive numbers.

2.2 Modify the method in Programming Project 2.1 so that the item with the highest key is not only returned by the method, but also removed from the array. Call the method `removeMax()`.

2.3 The `removeMax()` method in Programming Project 2.2 suggests a way to sort the contents of an array by key value. Implement a sorting scheme that does not require modifying the `HighArray` class, but only the code in `main()`. You'll need a second array, which will end up inversely sorted. (This scheme is a rather crude variant of the selection sort in Chapter 3, "Simple Sorting.")

2.4 Modify the `orderedArray.java` program (Listing 2.4) so that the `insert()` and `delete()` routines, as well as `find()`, use a binary search, as suggested in the text.

Q 5. In Lab Three, we compare the execution times of three sort algorithms with the same integer arrays. Similarly, in this question, write a java program to compare the execution times of the linear search and binary search we have learnt in the lectures. You can implement this by **adding the following two methods in bubbleSort.java**:

```
public  int linearSearch(long key);

public int binarySearch(long key);
```

In each method, if *key* is found in the array, return the corresponding array index; otherwise, return *nElems*. *linearSearch()* is similar to *find()* in highArray.java, in which *key* is searched sequentially; *binarySearch()* is similar to *find()* in orderedArray.java, in which *key* is searched based on the binary search method.

Measure the execution times of linearSearch() and binarySearch() with the same input *key (a random number)* based on **the same sorted array** (*generated with the methods in Lab Three*). Conduct experiments with different array sizes (for the array size with 10,000 or 100,000 integers, conduct the experiments for 5 times; for the case with 1,000,000 integers, run your program once as it may takes several hours) and fill the table as follows:

		Execution time (ns)	
		Linear Search	Binary Search
10,000 integers	1st		
	2nd		
	3rd		
	4th		
	5th		
100,000 integers	1st		
	2nd		
	3rd		
	4th		
	5th		
1,000,000 integers	1st		

Do the experimental results match what we have learnt in the lectures (The binary search is very fast with O(lg(n)) and the linear search is slow with O(n))?