ECE368 Fall 2016 Homework 7

IMPORTANT:

- Do NOT leave your name or Purdue ID on this homework.
- Write your homework security number at the TOP of EACH page.

Read and sign the ***Academic Honesty Statement*** that follows:

"In signing this statement, I hereby certify that the work on this exercise is my own and that I have not copied the work of any other student while completing it. I understand that, if I fail to honor this agreement, I will receive a score of zero for this exercise and will be subject to further disciplinary action."

`Homework security number:`

Please acknowledge any people who have helped you with this homework.

Question	Credits
1	
2	
3	

1. (30 points) A *k-ary* max-heap is a generalization of a binary max-heap in which each node in the heap has at most k children instead of two. In other words, it is an almost complete k-ary tree in which every node satisfies the max-heap property.

(a) Given the array `A[12] = [S, O, R, T, I, N, G, I, S, F, U, N]`, show how you will build the max-heap for k = 3 by inserting the elements of array A in succession. Show the snapshot of the max-heap after each element has been inserted and any heapify operations have been completed.

(b) If the max-heap is represented by an array A, describe how to find the parent and the k children (at most) of element `A[i]`. Assume the array starts at `A[0]`.

(c) What familiar sorting algorithm is k-ary HeapSort really performing for `k = 1`? Justify your answer.

2. (30 points) Bob, the builder, has a set N of n nuts and a set B of n bolts, such that each nut in N has a unique matching bolt in B. Unfortunately, the nuts in N all look the same, and the bolts in B all look the same as well. The only kind of comparison that Bob can make is to take a nut-bolt pair (a,b), such that $a \in N$ and $b \in B$, and test it to see if the threads of a are larger, smaller, or a perfect match with the threads of b. Describe an efficient algorithm for Bob to match up all the nuts in N with the corresponding bolts in B. What is the average running time of this algorithm in terms of nut-bolt comparisons that Bob must do?

This is a great interview question. It's probably easy to come up with the straightforward, inefficient solution. The non-trivial solution, on the other hand, requires some thinking. How about we apply the spirit of "partitioning" from quicksort here?

3. (40 points) As we discussed in class, stable sorting algorithms maintain the relative order of records with equal values. That is, a sorting algorithm is stable if whenever there are two records R and S with the same key and with R appearing before S in the original list, R will appear before S in the sorted list as well. Give a simple scheme to make any comparison-based sorting algorithm stable. How much extra memory will your scheme need?