Problem Set 3

This problem set is due in class on *Tuesday, March 6th* at the end of class.

There are four problems. **Each problem is to be done on a separate sheet (or sheets) of paper. Separate problems should not be stapled together.** Mark the top of each and every sheet with your name, 6.046J/18.410J, the problem number, your recitation section, your TA’s name, and the date. It is very important to hand in problems separately and to indicate your recitation section and TA name on every sheet. Failure to do so will result in delays in returning your work and possible confusions with entering your grade. All problem sets throughout the semester should be handed in this way.

**Reading:** Chapters 7–8, 9.1–9.3 of CLR.

**Problem 3-1. Heap Delete**

The operation $\text{HEAP-DELETE}(A, i)$ deletes the item in node $i$ from heap $A$. Give an implementation of $\text{HEAP-DELETE}$ that runs in $O(\lg n)$ time for an $n$-element heap.

**Problem 3-2. $k$-ary Heaps**

Ben Bitdiddle decides to extend the idea of a binary heap to a $k$-ary heap. Thus, each node in the heap now has $k$ children instead of just two.

(a) If the heap is represented by an array, describe how to find the parent and the $k$ children of element $A[i]$.

(b) Ben decides to implement heapsort using his $k$-ary heaps. He chooses $k = 1$ and argues that for this choice of $k$, the only operation required in the heapsort algorithm is $\text{BUILD-HEAP}$. Since $\text{BUILD-HEAP}$ takes $O(n)$ time, he can actually sort in $O(n)$ time! Find the fallacy in Ben’s argument. What familiar sorting algorithm is $k$-ary heapsort really performing for $k = 1$?

(c) For $k > 1$, analyze how many binary comparisons are used by $k$-ary heapsort in the worst case. Express your answer in terms of $n$ and $k$. What value of $k$ minimizes this expression?

**Problem 3-3. Pancake Sorting**

Alyssa P. Hacker has cooked a stack of $n$ pancakes, $p_1, p_2, \ldots, p_n$ and she would like to sort the stack so that the pancakes are ordered from smallest on top to largest on the bottom. Unfortunately, the only operation she can perform is to insert a spatula between two pancakes in the stack and “flip”. For example, if she inserts the spatula beneath pancake 1 and flips, she gets:
<table>
<thead>
<tr>
<th>Before flip</th>
<th>After flip</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Give an algorithm to solve Alyssa’s problem that uses only the flip operation to reorder pancakes. Make the algorithm as efficient as you can with respect to flips. Of course, Alyssa is allowed to examine the pancakes in the stack before choosing a location to insert the spatula. She can also easily recognize the smaller of any pair of pancakes she compares, even if they are not adjacent in the stack. How many flip operations does your algorithm take? In the worst case, how many comparisons does your algorithm require Alyssa to perform?

**Problem 3-4. Sorting Strings and Things**

(a) One day you walk too close to tech support, and a support person grabs you and asks you to answer a sorting question from some caller. So you get on the phone, and have the following conversation:

Caller: “I have this file I want to sort the lines of in alphabetical order, and I want to know how fast I can do it.”
You: “How many lines are there?”
Caller: “I don’t know.”
You: “Do you know know how long the longest line is?”
Caller: “No.”
Just then a 6.046 TA strides by and shouts out: “If the file has $n$ characters total, it can be done in $O(n)$ time.”
Caller: “Great! How do I do that?”
You better come up with an answer. Think quickly!

(b) The caller ponders your brilliant solution to part (a), and then says, “That’s great, but I have another file that has a positive integer on each line. If the total number of digits in the file is $n$, can I sort that in $O(n)$ time too?”
You: “That sounds like the same problem.”
You hear someone shout in the distance, “You need a different algorithm, but it’s still linear time in total digits!” (Don’t these TAs have anything better to do?) You better explain this one too.