**Fall 2006 CMSC 651: Homework 1 ORAL Due: Due Sep 14 or 15**

(see email about Oral HW)

READING: Notes on AMORTIZED DATA STUCTURES that are posted.

1. (25 points) Let $F$ be the Fib-heap consisting of $n$ isolated nodes, labelled with keys $\{1, 2, \ldots, n\}$, with $\{1, 2, \ldots, \lfloor n/2 \rfloor\}$ marked, and the rest unmarked. There is a pointer to the node with key 1 since it is minimum.

   (a) Imagine that a call is made to DELETEMIN. What does the Fib-heap look like now?

   (b) After that, imagine that a call is made to DECKEY to the node with key $\lfloor n/3 \rfloor$. What does the Fib-heap look like now?

2. (25 points) Let $a$ be a constant.

   - Define a sequence of trees $C_0, C_1, \ldots$ to be $C_0$ is a 1-node tree. For all $i \geq 0$, $C_{i+1}$ is formed by taking $a$ copies of $C_i$, and hanging $a-1$ of them off of the root of the $a$th one.
   - Define an $a$-NOMIAL HEAP to be just like a BINOMIAL HEAP except that
     - instead of $B_i$'s we use $C_i$'s,
     - instead of only allowing ONE of each $C_i$, we allow $a-1$ of each $C_i$.

   For an $a$-NOMIAL HEAP describe algorithms for UNION, INSERT, FINDMIN, DELETEMIN, DECKEY that are similar to those for BINOMIAL HEAP. Analyze how long each operation takes; however, keep track of how $a$ affects the number of steps. For example, it may be that DECKEY takes $O(a \log n)$ or $O(\log(n/a))$ steps where the constant in the 'O' is ind of $a$.

**GOTO PAGE 2- THERE IS A PAGE 2**

3. (25 points) (This problem is based on what one might *think* a Fib-heap is.) For this problem let $f_0 = 1$, $f_1 = 1$, and, for all $i \geq 1$, $f_{i+1} = f_i + f_{i-1}$ (the Fib Numbers).

   We define a data structure as follows.

   - $F_0$ is a 1-node tree. $F_1$ is a 2-node tree. For all $i \geq 1$, $F_{i+1}$ is formed by taking $F_i$ and hanging $F_{i-1}$ off of its root, or by taking $F_{i-1}$ and hanging off of its root an $F_i$. (So $F_{i+1}$ is a set of types of trees.)
   - Define a SAMIR-HEAP to be just like a BINOMIAL HEAP except that
     - instead of $B_i$'s we use $F_i$'s,
     - instead of NOT allowing TWO of any $F_i$, we DO NOT allow to have two $F$'s of adjacent indices. (e.g., do not allow an $F_3$ and an $F_4$– they can be made into an $F_5$.)
     - we maintain a pointer to the root that is minimal.

   (a) For SAMIR-HEAP describe algorithms for UNION, INSERT, FINDMIN, DELETEMIN, DECKEY that are similar to those for BINOMIAL HEAP. Analyze how long each operation takes.

   (b) Compare and contrast SAMIR-HEAPS, BINOMIAL HEAPS, and FIB-HEAPS for all the usual Data Structure operations we have encountered. If (say) two of the data structures both do the same asymptotically (e.g., if they both do DELETEMAX in $O(\log n)$ steps) then be prepared to discuss which one is faster. In other words, we may care about the contant in the O-of term.

4. (25 pts) Dr. Hcrasag thinks that Fib-heaps are too complicated! He thinks that an easier version that does not bother with the markings is just as good. He defines H-heaps to be just like Fib-heaps except for the following two changes.

   - No nodes are ever marked or unmarked.
   - In the DECKEY operation you just make the subheap rooted at the key you want to reduce into a new heap in the set of heaps.
   - The potential function is $n(H)$, the number of heaps.

   This DOES NOT WORK. This problem is about seeing where it goes wrong.

   (a) Find a sequence of $O(n)$ operations that results in some node having degree $>> \log n$.

   (b) Which operation or operations have a larger amortized cost in an H-heap than in an Fib-heap? Give example (or examples) of a sequence of operations where that operation has a large amortized cost.