CS 61B **Discussion 10** Spring 2015

1 Heaps of fun®

(a) Assume that we have a binary min-heap (smallest value on top) data structure called Heap that stores integers and has properly implemented insert and removeMin methods. Draw the heap and its corresponding array representation after each of the operations below:

```
Heap h = new Heap(5); //Creates a min-heap with 5 as the root
h.insert(7);
h.insert(3);
h.insert(1);
h.insert(2);
h.removeMin();
h.removeMin();
```

(b) Your friend Alyssa P. Hacker challenges you to quickly implement an integer max-heap data structure - "Hah! I'll just use my min-heap implementation as a template to write max-heap.java", you think to yourself. Unfortunately, your arch-nemesis Malicious Mallory deletes your min-heap.java file. You notice that you still have the min-heap.class file; could you use it to complete the challenge? – you can still use methods from min-heap but you cannot modify them. If so, describe your approach, do not write code. If not, explain why it is impossible.

2 HashMap Modification (from 61BL SU2010 MT2)

(a) When you modify a key that has been inserted into a HashMap will you be able to retrieve that entry again? Explain.
□ Always □ Sometimes □ Never

(b) When you modify a value that has been inserted into a HashMap will you be able to retrieve that entry again? Explain.
□ Always □ Sometimes □ Never

3 Sum Paths

Define a root-to-leaf path as a sequence of nodes from the root of a tree to one of its leaves. Write a method `printSumPaths(TreeNode root, int k)` that prints out all root-to-leaf paths whose values sum to k. For example, if RootNode is the binary tree rooted in 10 in the diagram below and k is 13, then the program will print out `10 2 1` on one line and `10 3 0` on another.

```
    10
   /  \
  2    3
 / \    \
5   1    0
```

Provide your solution by filling in the code below:

```
void printSumPaths(Node t, int k) {
    if (t != null) {
       sumPathsHelper(                                    );
    }
}
void sumPathsHelper(                                  ) {











}
```

Bonus question: What is the worst case $\Theta(\cdot)$ runtime of your method in terms of the number of nodes N in the given tree?

4 Bonus Question

Describe a way to implement a linked list of Strings so that removing a String from the list takes constant time. You may assume that the list will never contain duplicates.