University of Waterloo
CS240 Fall 2020
Midterm Post Mortem

Problem 1 [1+1+2=4 marks]

- For part a), a few students wrote incorrect runtimes (log n or n, instead of sqrt(n))
- For part c), some students gave incorrect depth of recursion or did not mention depth at all or gave incorrect # of calls or did not mention # of calls at all. Note here MSD doesn't do anything for input size 0 or 1.

Problem 2 [2+2+3=7 marks]

- For part a), some students left out average case height is $\log n$
- For part b), some students assumed linking of elements so total time $\log n$
- For part c), some students got wrong for lower bounds of e.g. $1, \log n$ for internal nodes. Note here an end-character branch must always lead to a leaf, so the lower bound cannot be treated as a complete ternary tree.

Problem 3 [2+1=3 marks (1 bonus mark)]

- For part a), students try to argue for the time to insert/delete each node, instead of using the contradiction argument
- Note: No assumptions can be made about how insert/deleteMax are implemented, and that the priority queue might not even resemble a heap.

Problem 4 [2+4=6 marks]

- For part a), some students had off by one errors, e.g. $\log_4(n+1)$
- For part b), students take the even parts to be n^2 as well.

Problem 5 [4+4=8 marks]

- For part a), many students gave an incorrect or missing exact expression.
- For part b), some students messes up with the first or second algorithm's runtime.

Problem 6 [7 marks]

- For part b), many students confused height and levels and were off by 1 as a result
- For part d), many students did not know how to use the harmonic numbers bound.

Problem 7 [3+1=4 marks]

- For part a), some students did not provide the skiplist.
- For part a), a few students incorrectly counting the number of levels instead of the number of vertical edges.
- For part b), almost everyone missed the +2 for sentinel.

Problem 8 [8 marks]

- A lot of students didn't take duplicate heights into consideration.
- Some students try to add one extra field without any additional data structure in skipList to store the lowest; the issue with this is that once lowest is deleted we cannot update this field in O(log(n)) time since they didn't keep track of other nodes with same height.
- Some students didn't store link from skiplist nodes to AVL/min-heap nodes, leading to a linear search in delete (duplicate items)
- Several students mentioned a delete(k) function for heaps without elaborating on it, but the standard heap design (as presented in lectures) does not contain any such function.
- Several students attempted to use an array or linked list as the auxiliary structure, but this would tend to cause either insert or delete (particularly when deleting the key with the lowest tower) to become slow (linear time) in the worst-case.