University of Waterloo
CS240 Spring 2023
Midterm Post-Mortem

This document goes over common errors and general student performance on midterm questions. We put this together using feedback from the graders once they are done marking. It is meant to be used as a resource to understand what we look at while marking and some common areas where students can improve in.

Question 1

- For part f), this statement would be true if everything was unique. However, without such assumption, this statement would be false.
- For part g), average running time may be equivalent to best case, but it should never be better than the best case. If it was true, best-case runtime is not best anymore.
- For part h), height of heap with n nodes is the shortest that any binary tree can go, due to property of heap. Simple intuition here is that AVL tree is okay with some empty spaces in specific level whereas heap is not.

Question 2

- This question was well done overall.
- For part b), some students chose incorrect c value for the Ω bound. It is important to recognize that the inequality must hold for **all** n greater or equal to n_0.
- For part d), some students were confused on how many buckets are needed. The key point was to notice the target base that we are converting into, which will affect number of digits and number of buckets.
- For part d), one common mistake was miscounting the number of digits at R = 2. The base 2 number needs at most 9 digits to represent 257.

Question 3

- We explicitly stated that you should use first principles - those work that used limit rule received deductions.
- Some students split $f(n)$ in two terms and proved that each term is less than $cg(n)$. One needs to be careful on how to split the term up - an approach to prove each term is less than $cg(n)$ may not be correct. If one proved that each term is less than $\frac{1}{2}cg(n)$ for example, then one might have got valid n_0.

Question 4

- This question was well done overall.
- Some students received deduction as their summation was not formed correctly or was lack of details.
- Some students counted the number of possible inputs or number of iterations incorrectly.

Question 5

- For part b) and c), we asked you to give us an array. Visualization of heap in tree form received a deduction.
- For part d), some students tried to give us the bound on number of comparisons, rather than actual number of comparisons. Such approach received deduction.

Question 6

- Some students did not analyze a decision tree and presented an possible algorithm. This is not accepted.
- Some students gave us one specific decision tree/algorithm and showed that chosen algorithm does at least 7 comparisons. The question's key was to prove that *any comparison based sorting algorithm* does at least 7 comparisons.
- Some students proved it using the fact that lower bound is in $\Omega(n \log n)$ and plugged $n = 5$ into $n \log n$. This is not a valid approach - we know that height is *in bound* of $n \log n$, not that it is actually $n \log n$.

Question 7

- With part a), some students performed one of double rotations, where the correct answer is to perform single rotation. In such situation where one can do both single or double rotations, ties must be broken to prefer single rotation.
- With part b), a lot of students did well on initial `restructure`. However, there were some students who made a mistake when fixing imbalance at the root after initial `restructure`.

Question 8

- In part a), some students answered that this function returns k^{th} *largest* items. `Partition` returns the number of items that are less than chosen pivot value.

- In part b), some students failed to consider the runtime of partition in their best case. This goes with part d) as well - some students did not consider cost of partition while when writing recurrence relation.
- In part d), some students failed to account for the $\theta(n)$ time partition function in the recursive case of the runtime.

Question 9

- The only data structure that you could have used for this question was AVL tree. Submissions using other data structure received deduction.
- Some students forgot to update their pointer to minimum and maximum node during/after `delete` operations or `insert` operations.
- Some students did not consider full possibility for next min/max nodes. For example with max node, it is possible that either its parent or its left child is next largest in AVL tree. You need to check both scenarios.
- Some students had augmented min and max pointer for each node. This would be at the end O(n) space, but this does require 2 extra fields for each node, which is ultimately O(n) more space required than usual AVL tree (i.e. key, value, height, and balance factor).
- Some students argued that locating min or max node in AVL tree is constant time (given that there is no pointer to each). This is not true, as you would have to traverse down either left or right side of AVL tree which takes $O(h)$ time.