University of Waterloo
CS240E, Spring 2026
Written Assignment 1 Post-Mortem

Question 1 [9 marks]

This question is well done.

Question 2 [3+3+3=9 marks]

Everyone who submitted Q2 got part (a) and (b) correct.

Some of you did not attempt part (c). The main idea is that at $n = 2^{2^k}$, the function is $f(2^{2^k}) = n$. At $n = 2^{2^k} - 1$, the function gets arbitrarily close to $f(2^{2^k} - 1) = 2^{2^{k-1}} \approx \sqrt{n}$.

For any $d < 1$, you can show that the function is not $O(n^d)$. For any $d > \frac{1}{2}$, the function isn't $\Omega(n^d)$.

Question 3 [7 marks]

Overall, this question is well done. Reminder that when an assignment asks you to "Describe an algorithm", you need to go through the 4-step process (idea, algorithm, correctness, runtime).

There are 2 common small mistakes in the proof of runtime:

1. As $n \to \infty$, the "worst-case" s can stay at 0. And no correct algorithm would run in $O(s) = O(0)$ time when $s = 0$. Therefore, the runtime is $O(1 + s)$, which is **not** $O(s)$.

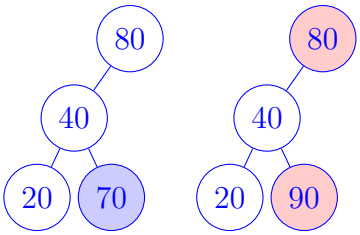
2. Some of you correctly identified that only s nodes are returned, and 1 node has to be visited at the start (which is the root). However, you failed to consider the $\leq 2s$ children nodes that are visited before being pruned by $key < x$. (Depending on the algorithm, this may or may not exist.)

Question 4 [3+7+3=13 marks]

Part (a) and (c) are mostly well done. Some of you didn't give a flagged tree for part (a).

Part (b)'s main idea is to biject the flagged tree to a multiway heap (briefly covered the intuition in Tutorial 2). By doing so, you realize that all right-child relationships become sibling relationships, and thus you only check left child relationships until the root.

There are a few common mistakes in part (b). One possible mistake is ending the increase-key check immediately after a right child relationship, which is too early. See the flagged tree on the following page: if you call increase-key(node with key 70, 90), and end the loop immediately since 90 is a right child, it violates the binomial heap ordering property.


80
40
20
70
80
40
20
90