University of Washington
Department of Computer Science and Engineering
CSE 417, Autumn 2023

November 3, 2023

Homework 6, Due Friday, November 10, 2023, 11:59 PM

Turn in instructions: Electronics submission on GradeScope. Submit as a PDF, with each problem on a separate page.

In the following problems on, you can ignore rounding issues (just round down to the nearest integer). A big-Oh answer is sufficient. You should solve these problems by unrolling the recurrence. Do not rely on the *master theorem.*

Problem 1 (10 points):
Solve the following recurrences:

a) $T(n) = 3T(n/2) + n$ for $n \geq 2$; $T(1) = 1$;

b) $T(n) = 16T(n/4) + n^2$ for $n \geq 2$; $T(1) = 1$;

c) $T(n) = 4T(n/3) + n^{3/2}$ for $n \geq 2$; $T(1) = 1$;

Problem 2 (10 points):
Solve the following recurrences.

a) $T(n) = T(n-1) + n$ for $n \geq 2$; $T(1) = 1$;

b) $T(n) = T(n/2) + 1$ for $n \geq 2$; $T(1) = 1$;

c) $T(n) = T(\sqrt{n}) + 1$ for $n > 2$; $T(2) = 1$; (You may also consider this recurrence to be $T(n) = T(\lfloor\sqrt{n}\rfloor) + 1$ to only have integer values.)

Problem 3 (10 points):
Let A and B be two sorted arrays of integers, each of length n. Show how you can find the median of the combined set of elements in $O(\log n)$ comparisons. (As in the Median algorithm discussed in lecture, you will need to solve the Select the k-th largest problem.) Justify your algorithm is correct.

Problem 4 (10 points):
Given an array of elements $A[1, ..., n]$, give an $O(n \log n)$ time algorithm to find a majority element, namely an element that is stored in more than $n/2$ locations, if one exists. Note that the elements of the array are not necessarily integers, so you can only check whether two elements are equal or not, and not whether one is larger than the other. HINT: Observe that if there is a majority element in the whole array, then it must also be a majority element in either the first half of the array or the second half of the array. (This is also exercise 3, page 246 from the text, without the annoying story line.)

Programming Problem 5 (10 points):

Leetcode 109. Convert Sorted List to Binary Search Tree.

Problem 6 (0 points) A cute problem, just for fun. Not graded:

Suppose you are working in the quality control of a factory that produces quarters for the US government and your job is to make sure that all quarters have exactly the same weight. You are given 2^k quarters for $k \geq 2$ and you know that at most one of them can be defective. A defective quarter will weigh higher or lower than normal. You are given a scale with two trays: Each time you can put a set S of quarters in the left and a set T in the right (for disjoints sets S, T). The scale will show if S is heavier than T, or T is heavier than S, or they have exactly the same weight. Design an algorithm to find the defective quarter (if it exists) by using the scale only $k + 1$ times. (Note that your algorithm will run by a human not a computer.) Justify your algorithm is correct.