University of Washington
Department of Computer Science and Engineering
CSE 417, Winter 2020

February 4, 2020

Homework 5, Due Wednesday February 12, 9:29 am, 2020

Turn in instructions: No need to turn these in, these are problems to help you gain familiarity with material for the midterm. These problems are not necessarily representative of the type of problems that will appear on the exam.

In the problems on this assignment, 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 (0 points):

Solve the following recurrences:

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

b) $T(n) = T(3n/4) + n$ for $n \geq 2$; $T(1) = 1$;

Problem 2 (0 points):

Solve the following recurrences:

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

b) $T(n) = 7T(n/3) + n^2$ for $n \geq 2$; $T(1) = 1$;

Problem 3 (0 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 $n \geq 2$; $T(1) = 1$;

Problem 4 (0 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.)