Design and Analysis of Algorithms
CS575, Spring 2025

Theory Assignment 2.1

Due on 3/6/2025 (Thursday)

Remember to include the following statement at the start of your answers with a signature by the side. "I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating, I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment for my first offense and that I will receive a grade of "F" for the course for any additional offense."

Please handwrite or type your answer to each question, scan or save your answers into a pdf file (with a vertical orientation so that we do not need to rotate your file to grade), and upload it to the homework submission site.

1. (16 points) Use the iteration method or recursion tree method to solve the following recurrence equation.
   a) (8 points)
   $$T(n) = \begin{cases} T(n-1) + n & if\ (n > 1) \\ 1 & if\ (n = 1) \end{cases}$$
   b) (8 points) You can assume $n^{1/2^k} = 2$ for some integers *n* and *k.*
   $$T(n) = \begin{cases} 0 & if\ n = 2 \\ T(\sqrt{n}) + 1 & if\ n > 2 \end{cases}$$

2. (6 points) Use Master method to solve $T(n) = 4T(n/2) + n^2$ and T(1)=1.

3. (10 points) Professor Caear wishes to develop a matrix-multiplication algorithm that is asymptotically faster than Strassen's Algorithm. His algorithm will use divide-and conquer method, dividing each matrix into pieces of size n/4× n/4, and the divide and combine steps together will take $\theta(n^2)$ time. He needs to determine how many subproblems his algorithm has to create in order to beat Strassen's algorithm. If his algorithm creates a subproblems, then the recurrence for the running time T(n) becomes T(n) = aT(n/4)+ $\theta(n^2)$. What is the largest integer value of a for which Professor Caesar's algorithm would be asymptotically faster than Strassen's algorithm?

4. (15 points) We want to find the largest item in a list of n items.
   a) Use the divide-and-conquer approach to write an algorithm (pseudo code is OK). Your algorithm will return the largest item (you do not need to return the index for it). The function that you need to design is *int maximum (int low, int high)*, which *low* and *high* stands for low and high index in the array. (8 points)

b) Analyze your algorithm and show its time complexity in order notation (using θ). (7 points)

5. (15 points) Illustrate the operation of Heapsort on the input array A = <5, 13, 2, 25, 7, 17, 20, 8, 4>. Draw the heap just after Build-Max-Heap was executed. Then draw a new heap after another (the next) element has been sorted; the last heap you draw has a single element (see example figure in slide 24 of Ch6-sorting-heap-linear lecture notes).

6. (15 points) Argue for the correctness of Heapsort (the slide 25 of Ch6-sorting-heap-linear lecture notes) using the following loop invariant: At the start of the iteration with an i of the for loop, (a) the subarray $A[1 .. i]$ is a max-heap containing the i smallest elements of $A[1 .. n]$, and (b) the subarray $A[i+1 .. n]$ contains the $n - i$ largest elements of $A[1 .. n]$ in correctly sorted order (i.e., in ascending order). Divide your proof into the three required parts: Initialization, Maintenance, and Termination.