Problem 2-1 (Sort recurrences) 8 Points

Sort the following recurrences in increasing order of growth of the corresponding functions. Justify (very) briefly.¹

(a) \( T(n) = 2T(n/4) + n; \)
(b) \( T(n) = 2T(n/3) + \log n; \)
(c) \( T(n) = 5T(n/3) + n^{1.9}; \)
(d) \( T(n) = 2T(n/2) + n; \)
(e) \( T(n) = T(n/10) + 3; \)
(f) \( T(n) = 27T(n/3) + 3n^3; \)
(g) \( T(n) = 4T(n/4) + n^2. \)

Problem 2-2 (Methods for Solving Recurrences) 12 points

Consider the recurrence \( T(n) = T(n/5) + T(n/2) + n. \)

(a) (4 Points) Using a recursion tree, determine a tight asymptotic upper bound on \( T(n). \)
(b) (4 Points) Prove your upper bound using induction.
(c) (4 Points) Using the substitution method, solve the recurrence \( U(n) = 3U([n^{1/3}]) + 7 \) with \( U(2) = 1. \)

Problem 2-3 (Functionality vs. Running Time) 10 points

Consider the following recursive procedure.

\[
\text{Bla}(n): \\
\text{if } n = 1 \text{ then return } 1 \\
\text{else return } \text{Bla}(n/3) + \text{Bla}(n/3)
\]

(a) (3 points) What function of \( n \) does Bla compute (assume it is always called on \( n \) which is a power of 3)?

¹For this entire homework assignment, you may ignore the fact that the argument to \( T \) may not be an integer.
(b) (3 points) What is the running time $T(n)$ of Bla (assuming the if statement and the addition can be accomplished in constant time)?

(c) (4 points) How do the answers to (a) and (b) change if the last line is replaced by

“else return $2 \cdot \text{Bla}(n/3)$”?

Problem 2-4 (Counting Inversions) 16 points

Let $A[1\ldots n]$ be an array of pairwise different numbers, where for simplicity you may assume that $n$ is a power of two. We call pair of indices $1 \leq i < j \leq n$ an inversion of $A$ if $A[i] > A[j]$. The goal of this problem is to develop a divide-and-conquer based algorithm running in time $\Theta(n \log n)$ for computing the number of inversions in $A$.

(a) (8 points) Suppose you are given a pair of sorted integer arrays $A$ and $B$ of length $n/2$ each. Let $C$ an $n$-element array consisting of the concatenation of $A$ followed by $B$. Give an algorithm (in pseudocode) for counting the number of inversions in $C$ and analyze its runtime. Make sure you also argue (in English) why your algorithm is correct.

(b) (8 points) Give an algorithm (in pseudocode) for counting the number of inversions in an $n$ element array $A$ that runs in time $\Theta(n \log n)$. Make sure you formally prove that your algorithm’s running time (e.g., write the recurrence and solve it.)

(Hint: Combine merge sort with part (a).)