Name:

Student ID #: 

- You are expected to work on this assignment on your own
- Use pseudocode, Python-like or English to describe your algorithms. Absolutely no C++/C/Java
- When designing an algorithm, you are allowed to use any algorithm or data structure we explained in class, without giving its details, unless the question specifically requires that you give such details
- Always remember to analyze the time complexity of your algorithms
- Homework has to be submitted electronically via Gradescope by the deadline. No late assignments will be accepted
Problem 1. (25 points)
Given the following recurrence relation

\[ T(n) = \begin{cases} 
1 & n = 1 \\
T\left(\frac{n}{9}\right) + \sqrt{n} & n > 1 
\end{cases} \]

1. Solve it exactly (i.e., without using any asymptotic notation) by iterative substitutions

2. Prove by induction that your exact solution is correct (do not prove a bound, but the exact solution)

Answer:
Problem 2. (25 points)

Using the Master method, give an asymptotic tight bound for $T(n)$ in the following recurrence relation

$$
T(n) = \begin{cases} 
    1 & n = 1 \\
    T \left( \frac{n}{3} \right) + n \log_3 n & n > 1 
\end{cases}
$$

Answer:
Problem 3. (25 points)

Suppose that we have designed three divide-and-conquer algorithms that solve a particular problem, where the input size is $n$. The first one solves four subproblems of size $n/2$ and the cost of combining the solutions of the subproblems to obtain a solution for the original problem is $n^2$. The second solves three subproblems of size $n/2$ and requires $n^2 \sqrt{n}$ time for combining the solutions. The third solves five subproblems of size $n/2$ and requires $n \log n$ time for combining the solutions. Assume that all three take $\Theta(1)$ when $n = 1$. Which algorithm would you choose and why? Show your work using the Master method.

Answer:
Problem 4. (25 points)

The median of a set of numbers \( \{a_1, a_2, \ldots, a_n\} \) is the element \( a_i \) such that there are \( \lceil n/2 \rceil \) elements smaller than or equal to \( a_i \), and there are \( \lfloor n/2 \rfloor \) greater than or equal to \( a_i \). In other words, the median is the element in the middle when the elements are sorted. For example, the median of \( \{7, 3, 4, 1, 9, 2, 13\} \) is 4.

You are given two sorted arrays \( A \) and \( B \) of size \( n \) each (for simplicity, you can assume \( n \) to be some power of 2 and that the numbers are distinct). Give an algorithm to find the median of all \( 2n \) numbers in \( O(\log n) \) time.

Answer: