1) Let
\[ T(n) = \begin{cases} 
1 & \text{if } n=1 \\
T\left(\left\lfloor n / 5 \right\rfloor\right) + T\left(\left\lfloor n / 4 \right\rfloor\right) + T\left(\left\lfloor n / 3 \right\rfloor\right) + O(n) & \text{if } n>1
\end{cases} \]
Prove by constructive induction that \( T(n) \) is \( O(n) \).

2) Explain how to make quick-sort run in worse case \( O(n \log n) \) time. Explain why I named this modified algorithm “slow-sort” in class.

3) Let \( n \) be a power of 2. Give an exact lower bound on the number of comparisons that merge-sort uses to sort \( n \) elements, in the worst case.
(HINT: Assume that the solution to problem 8-6 (2n-1) is an exact lower bound).

4) exercise 8.4-2
What is the worst-case running time for the bucket-sort algorithm? What simple change to the algorithm preserves its linear expected running time and makes its worst-case running time \( O(n \log n) \)?

5) Prove that \( x = \text{median}(x_1, x_2, \ldots, x_n) \) (where \( x_i \leq x_{i+1} \)) is a value that minimizes
\[ \text{SAD}(x) = |x_1-x| + |x_2-x| + \ldots + |x_n-x|. \]
(SAD=Sum of Absolute Differences)

HINT: rewrite
\[ \text{SAD}(x) = (2i-n)x - (x_1 + x_2 + \ldots + x_i) + (x_{i+1} + x_{i+2} + \ldots + x_n) \text{ when } x_i \leq x \leq x_{i+1}. \]

Why cannot we say “\( x = \text{median}(x_1, x_2, \ldots, x_n) \) is the value that minimizes \( \text{SAD}(x) \)”?