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 on Gradescope by the deadline. No late assignments will be accepted
Problem 1. (25 points)

A CS 141 student has been trying to speed-up Karatsuba’s divide-and-conquer integer multiplication algorithm. Given two numbers $x, y$ with $n$ bits each, her algorithm (1) first divides both $x$ and $y$ into four equal-length pieces, then (2) expresses the product $x \cdot y$ using $p$ multiplications of these $n/4$-bit pieces, followed by a constant number of additions, subtractions and shifts. How small $p$ needs to be in order to give a faster algorithm than the Karatsuba’s algorithm covered in class? You can assume $n$ to be a power of 4, and $p > 4$. Justify your answer.

Answer:
Problem 2. (25 points)

Give a divide-and-conquer algorithm ("Karatsuba-like") for multiplying two polynomials of degree $n$ in time $O(n^{\log_2 3})$.

Answer:
Problem 3. (25 points)

Describe and analyze an algorithm that takes an unsorted array $A$ of $n$ integers (in an unbounded range) and an integer $k$, and divides $A$ into $k$ equal-sized groups, such that the integers in the first group are lower than the integers in the second group, and the integers in the second group are lower than the integers in the third group, and so on. For instance if $A = \{4, 12, 3, 8, 7, 9, 10, 20, 5\}$ and $k = 3$, one possible solution would be $A_1 = \{4, 3, 5\}, A_2 = \{8, 7, 9\}, A_3 = \{12, 10, 20\}$. Sorting $A$ in $O(n \log n)$-time would solve the problem, but we want a faster solution. The running time of your solution should be bounded by $O(nk)$. For simplicity, you can assume that is $n$ a multiple of $k$, and that all the elements are distinct. Note: $k$ is an input to the algorithm, not a fixed constant.

Answer:
Problem 4. (25 points)

In the algorithm Select described in class (linear-time selection), the input elements are divided into \( n/5 \) groups of 5.

1. Suppose you modify the algorithm to divide the input elements into \( n/7 \) groups of 7 instead. Let \( T(n) \) denote the worst-case running time of the modified algorithm as a function of the input size \( n \). Write a recurrence relation for \( T(n) \), then give a proof that \( T(n) \in O(n) \).

2. Suppose you modify the algorithm to divide the input elements into \( n/3 \) groups of 3 instead. Let \( T(n) \) denote the worst-case running time of the modified algorithm as a function of the input size \( n \). Write a recurrence relation for \( T(n) \), then provide an argument that \( T(n) \) is not \( O(n) \).

Answer: