Problem 1. (30 points)
Give a divide-and-conquer algorithm for multiplying two polynomials of degree \( n \) in time \( O(n^{\log_2 3}) \).

Problem 2. (40 points)
The square of a matrix \( A \) is its product with itself, \( AA \).

- Show that five multiplications are sufficient to compute the square of a \( 2 \times 2 \) matrix.
- What is wrong with the following algorithm for computing the square of a \( n \times n \) matrix?

  “Use a divide-and-conquer approach as in Strassen’s algorithm, except that instead of getting 7 subproblems of size \( n/2 \) we now get 5 subproblems of size \( n/2 \) thanks to the previous observation. Using the same analysis as in Strassen’s algorithm, we can conclude that the new algorithm runs in time \( O(n^{\log_2 5}) \).”
Problem 3. (30 points)

Given an array $A$ of $n$ (possibly negative) integers, find two indices $1 \leq i \leq n$ and $1 \leq j \leq n$ such that the value of $\sum_{k=i}^j a_k$ is maximized.

Here some examples (the solution is underlined):

- $A = [-2, 11, -4, 13, -5, 2]$ which has answer 20,
- $A = [1, -3, 4, -2, -1, 6]$ which has answer 7,
- $A = [-1, 4, -3, 5, -2, -1, 2, 6, -21]$ which has answer 11.

Write an $O(n \log n)$-time divide and conquer\footnote{A $O(n)$ dynamic programming algorithm for this problem exists, but here you are supposed to give the slower divide and conquer algorithm.} algorithm for the problem described above. The algorithm should return $i$ and $j$. If all elements of the array are negative, the algorithm should return $i = j = 0$. 