1. Suppose you are choosing between the following three algorithms:
   - Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
   - Algorithm B solves problems of size $n$ by recursively solving two subproblems of size $n - 1$ and then combining the solutions in constant time.
   - Algorithm C solves problems of size $n$ by dividing them into nine subproblems of size $n/3$, recursively solving each subproblem, and then combining the solutions in $O(n^2)$ time.

   What are the running times of each of these algorithms and which one would you choose?

2. Write a pseudocode for the brute-force method of solving the maximum-subarray problem. What’s the complexity of your procedure?

3. Use Strassen’s algorithm to compute the matrix product $C = AB$, where $A = \begin{bmatrix} 1 & 3 \\ 7 & 5 \end{bmatrix}$ and $B = \begin{bmatrix} 6 & 8 \\ 4 & 2 \end{bmatrix}$. Show all intermediate results!

4. How many lines does the following program print, as a function of $n = 2^d$?
   ```c
   function fun(n)
   if n > 1
       print.line('good day')
       fun(n/2)
       fun(n/2)
   end if
   ```

5. A $k$-way merge problem. Suppose you have $k$ sorted arrays, each with $n$ elements, and you want to combine them into a single sorted array of $kn$ elements.
   (a) Here’s one strategy: using the `Merge` in the `MERGESORT` to merge the first two arrays, and then merge in the third, then merge in the fourth and so on. What is the time complexity of this algorithm, in terms of $k$ and $n$?
   (b) Give a divide-and-conquer algorithm to solve the $k$-way merge problem, and show the complexity of your algorithm.

6. The standard algorithm for multiplying two $n$-bit binary integers $x$ and $y$ costs $\Theta(n^2)$. A naive divide-and-conquer algorithm is to let $x = 2^{n/2}a + b$ and $y = 2^{n/2}c + d$, then

   $$xy = (2^{n/2}a + b)(2^{n/2}c + d) = 2^nac + 2^{n/2}(ad + bc) + bd$$

   The complexity is $T(n) = 4T(n/2) + \Theta(n) = \Theta(n^2)$. There is no improvement to the standard algorithm.
   (a) By observing that $ad + bc = (a + b)(c + d) - (ac + bd)$, we can use only three multiplications. Describe this divide-and-conquer algorithm in a pseudocode.
   (b) What is the complexity of the algorithm.
   (c) Illustrate the algorithm for multiplying integers $x = 10011011$ and $y = 10111010$ (note: just show one level of the recursion).