

Homework Assignment 1

Handed Out: Jan 6

Due: Jan 13

1. (10 pts) Suppose $T_1(n) = O(g(n))$ and also $T_2(n) = O(g(n))$. Which of the following are true? Explain your answer.
 - (a) $T_1(n) + T_2(n) = O(g(n))$.
 - (b) $T_1(n) - T_2(n) = o(g(n))$.
 - (c) $T_1(n)/T_2(n) = O(1)$.
 - (d) $T_1(n) = O(T_2(n))$.
2. (10 points) In a court order case, a judge cited a city for contempt and ordered a fine of \$10 for the first day. Each subsequent day, until the city followed the judge's order, the fine was **doubled**. (That is, the fine progressed as follows: \$10, \$20, \$40, etc.)
 - (a) (3 pts) What would be the fine on day N ?
 - (b) (3 pts) Given a specific amount D , how many days would it take for the fine to reach D dollars? (A big-Oh answer will do.)
 - (c) (4 pts) Suppose the judge were a bit harsher, and the fine was **squared** each day, instead of doubled. Then, what would be the fine on day N ?
3. (10 pts) You are given an $N \times N$ matrix of numbers, which is already in memory. The matrix is *monotone* in the following sense: in each row, the numbers are increasing from left to right; and in each column, the numbers are increasing from top to bottom. Give a worst-case $O(N)$ time algorithm to decide if a given number X is in the matrix.
4. (10 pts) You are given a sequence of n **rational numbers**, which are all positive, but not necessarily greater than 1. We want to find a subsequence with the **maximum product value**. For instance, in the input sequence 2, 1/10, 100, 1/4, 6, 1/10, 8, optimal subsequence is 100, 1/4, 6.

Describe the most efficient algorithm you can (along with an analysis of its running time) for this problem.
5. (10 pts) Given a sequence of n integers (positive and negative), we wish to find the **maximum average subsequence**. That is, the subsequence should have the largest possible *average* value over all possible subsequence, where the average of a subsequence is defined as its sum divided by its length. Describe an $O(n)$ time algorithm for this problem.

Next, suppose the subsequence is required to be at least ℓ numbers long, for some $1 \leq \ell \leq n$. Describe an $O(n^2)$ time algorithm to find the maximum average subsequence of length at least ℓ .