Assignment 1


1) Develop a Dynamic Programming algorithm which takes as input non-negative integers \(-a_1, a_2, \ldots, a_n; c_1, c_2, \ldots, c_n; b\) and returns the answer to
\[
\text{Max}_{x_i \geq 0, \text{integer}} \sum_{i=1}^{n} c_i x_i \quad \text{subject to} \quad \sum_i a_i x_i \leq b.
\]
It should run in time \(O(nb^2)\).

2) Suppose you are given \(n\) positive integers \(a_1, a_2, \ldots, a_n\). You are to find the greatest common divisor of the set of \(\binom{n}{n/2}\) integers, each obtained as the product of a subset of \(n/2\) of the \(n\) input integers. Develop a polynomial time algorithm for this problem.

3) You are given a directed graph \(G(V, E)\) with two vertices - \(s, t\) specified. You are to find the number of “walks” of length \(m\) from \(s\) to \(t\), where a walk is a sequence of edges so that the tail of each edge is the head of the next. [Edges may be repeated.] Devise an algorithm for this problem which runs in time polynomial in \(n, m\) (\(n = |V|\)).

Think about the problem of counting the number of simple paths from \(s\) to \(t\) of length \(m\), where a simple path is a walk with no edge being used more than once. You do not have to come up with a polynomial time algorithm for this, but describe in 5 or less lines why you cannot modify your algorithm for the first part to do this too.

4) Given a string of letters from a finite alphabet, a subsequence (of the string) consists of a subset of not necessarily contiguous letters in order from the string. [In contrast, the word “sub-string” denotes that we are taking a contiguous subset of letters.] Either think of or read up from any standard textbook a polynomial time Dynamic Programming algorithm for finding the longest common sub-sequence of two given strings.
Now develop a poly time algorithm for finding the longest common subsequence of three given strings.