1. Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is \(\langle 5, 10, 12, 5, 50, 6 \rangle\).

Simplified version of Ex. 15.2-1 of [CLRS] (2nd and 3rd eds)

2. Consider the problem of taking a set of \(n\) items with sizes \(s_1, \ldots, s_n\), and values \(v_1, \ldots, v_n\) respectively. We assume \(s_i, v_i \in \mathbb{N}\) for all \(1 \leq i \leq n\). Suppose we are also given a “knapsack capacity” \(C \in \mathbb{N}\). The knapsack problem is the problem of finding a subset \(S \subseteq \{1, \ldots, n\}\) such that \(\sum_{i \in S} s_i \leq C\) and such that \(\sum_{i \in S} v_i\) is maximized subject to the first constraint.

We write \(kp_{n,C}\) to denote the value \(\sum_{i \in S} v_i\) of the maximum-value knapsack on the set of all items. For any \(k \leq n\), and any \(\hat{C} \leq C, \hat{C} \in \mathbb{N}\), we can consider the same problem on the first \(k\) items in regard to capacity \(\hat{C}\). We denote the maximum-value knapsack for such a subproblem by \(kp_{k,\hat{C}}\).

(a) Prove that the following recurrence holds:

\[
kp_{k,\hat{C}} = \begin{cases} 
0 & \text{if } k = 0 \\
kp_{k-1,\hat{C}} & \text{if } k > 0 \text{ but } s_k > \hat{C} \\
\max\{kp_{k-1,\hat{C}}, kp_{k-1,\hat{C}-s_k} + v_k\} & \text{otherwise.}
\end{cases}
\]

(b) Use the recurrence in (a) to develop a \(\Theta(n \cdot C)\) dynamic programming algorithm to compute the optimal knapsack wrt the original \(n\) items and capacity \(\hat{C}\).

3. Longest Common Subsequence A subsequence of a given sequence is just the given sequence with some elements (possibly none) left out. Given a sequence \(s = s_1s_2 \ldots s_n\), we say another sequence \(r = r_1 \ldots r_k\) is a subsequence of \(s\) if there is a strictly increasing sequence \(i_1, i_2, \ldots, i_k\) of indices such that for all \(j = 1 \ldots k\) we have \(r_j = s_{i_j}\).

Given two sequences \(x\) and \(y\) we say that a sequence \(r\) is a common subsequence if \(r\) is a subsequence of both \(x\) and \(y\). In the longest common subsequence problem, we are given two sequences \(x = x_1 \ldots x_n\) and \(y = y_1 \ldots y_m\) and wish to find a maximum-length common subsequence of \(x\) and \(y\).

Give an \(O(mn)\)-time dynamic programming algorithm to solve the longest common subsequence problem.