Let $C(n)$ be the number of possible orders for performing the matrix multiplications $M_1 \times \cdots \times M_n$. The number is determined by the following recurrences: $C(1) = 1$ and $C(n) = \sum_{k=1}^{n-1} C(k)C(n-k)$, when $n \geq 2$. (Here’s why: The ’×’ that is performed last can be between matrices $M_k$ and $M_{k+1}$ for each $k = 1, \ldots, n-1$. The sequence $M_1 \times \cdots \times M_k$ can be evaluated in $C(k)$ different orders, and the sequence $M_{k+1} \times \cdots \times M_n$ can be evaluated in $C(n-k)$ different orders.)

1. Present a recursive algorithm based directly on the above recurrences, to compute $C(n)$. Estimate its complexity. (Hint: Write down the recurrence for $T(n)$, simplify it, and examine the difference $T(n) - T(n-1)$.)

2. Present a dynamic programming algorithm for computing $C(n)$. Analyze its complexity.

3. The dynamic programming algorithm that we discussed for optimizing a sequence of matrix multiplications computes only the smallest number $m_{1n}$ of MUL operations required. Extend the solution so that it also outputs an optimal ordering for the matrix multiplications using parentheses, for example as follows:

\[
(M_1 \times (M_2 \times M_3)) \times M_4
\]

Does this extension increase the asymptotic complexity of the solution? (Hint: Record the locations of ’×’ operations that lead to the optimum, and then use them and ”divide-and-conquer” to print an optimal parenthesizing.)

4. Extend our dynamic programming solution for the Knapsack problem with a procedure which, after tabulating the values, traces back and outputs the contents of an optimal load (as indices of its items). Argue that this does not increase the asymptotic $O(nT)$ complexity of the solution.

5. (a) Draw a decision tree for executions of insertion sort, when the keys to be sorted are given in array $A[1..3] = [a, b, c]$. Label each leave with the corresponding sorted order produced by the algorithm.

(b) Consider a decision tree for insertion sort, when there are $n$ keys to be sorted. Sketch the structure of the tree: How many leaves are there? Estimate the largest, the smallest, and the average depth of the leaves of the tree.