Dynamic Programming paradigm

- For any DP problem, we can define a formula (/several formulas) to describe the solution of the problem.
  This formula takes into account some general (inductive) cases and some base cases.
- Just as in the case of Greedy, we can define subproblems and the Optimal Substructure Property.
  Subproblems may contain more information than in case of Greedy.

An example of identifying these elements in case of the Rod Cutting Problem:

Input: n, $price[1 \dots n]$ where $price[i] \in \mathbb{N}$
a piece of length i is sold for $price[i]$ money.

Output: $maxProfit \in \mathbb{N}$, such that for $maxProfit$: is the largest number such that $\exists\ k, l_1, l_2, \dots l_k$ with:

- $\sum_{i=1}^{k} l_i = n$ (sum of lengths is total length n)
- $\sum_{i=1}^{n} price[l_i] = maxProfit$ (sum of selling prices is $maxProfit$)

I. The Subproblems: The instances of the problem in which we have to cut a rod of size x (in range {0,1,…n-1,n}).
II. The Optimal Substructure Property: Given l1+l2+…+lk an optimal cut (that brings maximum profit) for a rod of size n, then l2+…+lk is an optimal cut for a rod of size n-l1.

III. The formula for the rod cutting problem:

1. Base case: set $profit[0]$ to 0
2. General case: $profit[n] = \max\left\{ price[i] + profit[n-i] \mid i \in \overline{1,n} \right\}$