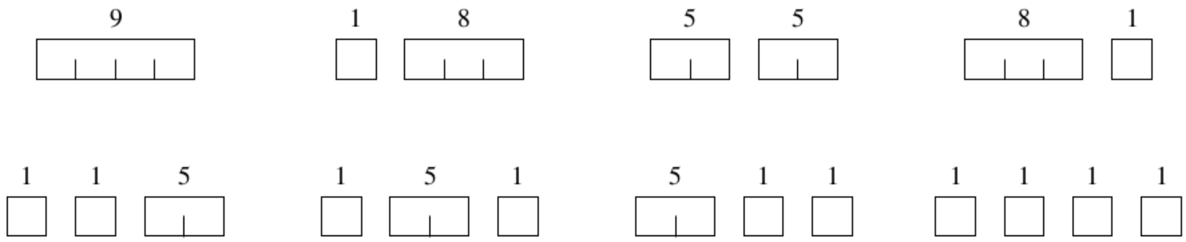
Exercises: Dynamic programming, first problem
Rod cutting problem

Example: *[Using the first 8 values from the example in the book.]*

length i	1	2	3	4	5	6	7	8
price p_i	1	5	8	9	10	17	17	20

Can cut up a rod in 2^{n-1} different ways, because can choose to cut or not cut after each of the first $n-1$ inches.

Here are all 8 ways to cut a rod of length 4, with the costs from the example:



The best way is to cut it into two 2-inch pieces, getting a revenue of $p_2 + p_2 = 5 + 5 = 10$.

1) For lengths i = 1, 2, …, 6 figure out the optimal way to cut the rod and the revenue r_i :

i	r_i	optimal solution
1	1	1 (no cuts)
2	5	2 (no cuts)
3	8	3 (no cuts)
4	10	$2 + 2$
5	13	$2 + 3$
6	17	6 (no cuts)
7	18	$1 + 6$ or $2 + 2 + 3$
8	22	$2 + 6$

2) Show, by means of a counterexample, that the following "greedy" strategy does not always determine an optimal way to cut rods. Define the ***density*** of a rod of length i to be p_i/i, that is, its value per inch. The greedy strategy for a rod of length n cuts off a first piece of length i, where $1 \le i \le n$, having maximum density. It then continues by applying the greedy strategy to the remaining piece of length $n - i$.

Example: p_1 = p_2 = 0, p_3 = 4, p_4 = 5

Optimal: take 1 piece of size 4, with p_4 / 4 = 5/4 = 1.25. Greedy first selects a piece of size 3, with p_3 / 3 = 4/3 = 1.33...

3) Can you write a recursion for computing the optimal revenue r_n for cutting a rod of size n in terms of p_i for $i = 1, \ldots, n$ and r_i for $i = 0, \ldots, n-1$? This is called **optimal substructure**.

Can determine optimal revenue r_n by taking the maximum of

- p_n: the revenue from not making a cut,
- $r_1 + r_{n-1}$: the maximum revenue from a rod of 1 inch and a rod of $n-1$ inches,
- $r_2 + r_{n-2}$: the maximum revenue from a rod of 2 inches and a rod of $n-2$ inches, ...
- $r_{n-1} + r_1$.

That is,

$$r_n = \max \{p_n, r_1 + r_{n-1}, r_2 + r_{n-2}, \ldots, r_{n-1} + r_1\} .$$

Optimal substructure: To solve the original problem of size n, solve subproblems on smaller sizes. After making a cut, two subproblems remain. The optimal solution to the original problem incorporates optimal solutions to the subproblems. May solve the subproblems independently.

A simpler way to decompose the problem: Every optimal solution has a leftmost cut. In other words, there's some cut that gives a first piece of length i cut off the left end, and a remaining piece of length $n - i$ on the right.

- Need to divide only the remainder, not the first piece.
- Leaves only one subproblem to solve, rather than two subproblems.
- Say that the solution with no cuts has first piece size $i = n$ with revenue p_n, and remainder size 0 with revenue $r_0 = 0$.
- Gives a simpler version of the equation for r_n:

$$r_n = \max \{p_i + r_{n-i} : 1 \le i \le n\} .$$