Algorithms. Assignment 3

When you describe dynamic programming algorithms in the following, make sure that you provide all ingredients: Define an "OPT function" (specify what its arguments and function value mean), then specify how you compute this OPT function and how you get an optimal solution, and analyze the time.

Problem 5

Problem 3 revisited: As the greedy attempts have failed, give now a dynamic programming algorithm for this problem.

This was pretty straightforward dynamic programming. Here is a more complicated example:

Problem 6

We are given n jobs. The ith job has size s_i, deadline d_i, and value v_i. All these numbers are positive integers. Here, "size" means that it takes s_i time units to execute the ith job. Once a job has started, its execution must not be interrupted. Furthermore, the time intervals where jobs are executed must be pairwise disjoint (but the end point of an interval may equal the start point of the next interval). Not all jobs must be done, but if the ith job is executed at all, it must end at its deadline d_i at the very latest. Hence it must start at a time no later than $d_i - s_i$.
The problem is to select a subset of jobs and to schedule them, in such a way that all selected jobs are finished before their deadlines, and the sum of values v_i of the selected jobs is maximized. The schedule starts at time 0.

As a "playful" application example, imagine that you want to watch videos from a multimedia database. Each video has a known duration and a subjective value (measuring how important it is for you), but unfortunately they are provided only for limited periods and will be removed after their respective deadlines.

(See next page.)

(a) We claim that the special case when all deadlines are equal is just the Knapsack problem. Motivate this statement. This might be obvious, but explain how the parameters of the two problems are related.

(b) Back to the general problem with different deadlines: Prove that there always exists an optimal solution such that the selected jobs appear in the same order as their deadlines. (For any two selected jobs, the job with earlier deadline is scheduled earlier.) Hint: Consider any optimal solution and re-order the jobs by a careful exchange argument. A slight difficulty is that the jobs have, in general, different sizes.

(c) Finally, use the property from (b) to design a dynamic programming algorithm, and analyze its time.

Special remark: Did you get a time bound that is polynomial in n? Then you should become wary: Is that possible? According to (a), the Knapsack problem is a special case ...