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 are supposed to mean), then specify how you compute this OPT function and how you get an optimal solution, argue why your calculation formula is correct, 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 standard dynamic programming. Here is an example that is less straightforward and might be challenging.

Problem 6

As earlier, assume that n houses $H_1, \ldots, H_n$ are located (in this order) along a straight road. We treat the road as the real line, such that every point has a real number as coordinate. Let s_i denote the point where house H_i is located; note that $s_1 < \ldots < s_n$. A vehicle stands at some point s at time 0. The vehicle must visit all houses (e.g., in order to bring some goods to customers living there). However, every house H_i must be visited before some individual deadline $d_i > 0$. Nothing special is assumed about these deadlines. The vehicle can drive at maximum speed 1, that is, traverse at most one length unit per time unit. (We also assume that the distances are large, such that we can neglect the acceleration and slow-down phases and the time for handing over the goods. We simply assume that the vehicle can instantly change its velocity, and a visit itself costs no time.)
A problem instance is given by the numbers $s_1, \ldots, s_n; d_1, \ldots, d_n$ and s, and the problem is to compute a route for the vehicle so as to reach every house H_i before the time d_i.
Beware: It is in general not enough to visit first all houses to the left of s, and then all houses to the right, or vice versa. It may be the case that costumers with close deadlines are waiting on both sides of s, and then you have to go forth and back. A complicated zigzag route may be needed. You can hardly avoid dynamic programming for this problem.

But here is a helpful simple observation: At every moment, the points that have already been visited form an interval.
There are several ways to set up the OPT function and, of course, this choice will influence your algortihm and time bound. If it helps, you may assume that all s_i and d_i are integers, and achieve a time bound that depends somehow on the largest $|s_i|$ and $|d_i|$. But if you are ambitious, you should aim for an algorithm whose time complexity is merely a nice polynomial in n, regardless of the numbers s_i and d_i.