Stockholm University
Department of Mathematics
Lecturer: Marc Hellmuth
Tutor: Anna Lindeberg and Liam Cuclair

2. Exercise "Algorithm and Complexity (DA 4005)"

This exercise counts as a part of the Practical Exercises

Exercise : Dynamic Programming

What a great party! You meet N friends, have a couple of drinks and enjoy the evening. One of your friends often comes up with funny questions and asks: *Can we align a couple of our smartphones so that the entire length is precisely* 1 *meter?* I mean, yes, this is one of your strange friends, but you are even stranger or should I say smarter ;) and you say: *I have taken this course on algorithms and we discussed the idea of dynamic programming. Wait . . . I know a way such that we can efficiently compute a solution, if one exists!* As teacher of this course, I can say, I am very proud of you - cheerio!

Explain and implement your solution. To be more precise, you have N smartphones $x_1, \ldots, x_N$ each of length $l(x_i) = l_i$ measured in cm. We can assume that the lengths are pairwise distinct and that they are only of the form "$x.y$"cm, where y is precisely one digit integer and x an integer less that 100cm (but of course greater than 0). The input to your program should be the lengths l_i, $1 \leq i \leq n$ (but its up to you if lengths should be provided in cm, mm or m) and the output of your program should be a possible solution, that is, specified smartphones for which their length sums up to 1meter (when such a solution exists) and "false"(otherwise).

To this end, design a dynamic programming approach as in the course, that is, you should

(a) Characterize the structure of an (optimal) solution.
(b) Recursively define the value of an (optimal) solution.
(c) Compute the value of an (optimal) solution.
(d) Construct an (optimal) solution from computed information.

Moreover, provide a pseudocode of your method. Your method should run for an arbitrary number N of smartphones in $O(N \times 1000)$ time and definitely not in $O(2^N)$ time. Provide a short proof for the time-complexity of your method.

Implement your algorithm in `C++`, `Java`, or `Python`. Turn in your code and the output of your program for $N = 7$ smartphones of length (in cm) $11; 12.9; 12.1; 14; 24.4; 24.6; 26$ and for $N = 6$ smartphones of length (in cm) $11; 12.9; 12.1; 14; 24.4; 24.6$.

When handing in programming exercises, always document how to compile and run your program. When submitting multiple files, hand them in a single zip-file. Comment your source code and do not copy from WWW!

Deadline: Wednesday, Oct 8 2025