University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
M. Fuchs, G. Schmid

Algorithms and Datastructures
Summer Term 2024
Grading Guidelines Exercise Sheet 11

Due: Wednesday, July 3rd, 4pm

Exercise 1: Wood Cutting *(8 Points)*

Given a wooden rod of length n and an array p of prices for selling a rod of some certain length, i.e., for $i \in \{1, \ldots, n\}$ we denote the price of a rod of length i by $p[i]$. Your task is to determine the *maximum value obtainable* by cutting up the given rod and selling the pieces according to the prices in p. For example, if the length of the given rod is $n = 5$ and the prices are as given in the following, then the maximum obtainable value is 26, by cutting the rod into two pieces of lengths 1 and one piece of length 3.

$$p[1] = 5, \quad p[2] = 8, \quad p[3] = 16, \quad p[4] = 19, \quad p[5] = 25$$

a) Let $OPT(n)$ be the maximal obtainable value for a rod of size n. Give a recursive formular on how to compute $OPT(n)$. *(4 Points)*

b) Give an algorithm that solves the problem efficiently. What is the runtime of your algorithm? *(4 Points)*

Sample Solution

First note that we can solve the problem in recursive manner. Let $OPT(i)$ be the optimal solution for a rod of size i:

$$\begin{aligned} OPT(0) &:= 0 \\ OPT(i) &:= \max_{j \in \{0,\ldots,i-1\}} \{OPT(j) + p[i-j]\} \end{aligned}$$

The algorithms' implementation works as follows: We use an array of size $n+1$ that stores $OPT(i)$ at index i. Clearly, filling the index at 0 takes constant time. To fill the array at position i, we have to look at every previous entry $0, \ldots, i-1$. This requires $O(i)$ time steps. Hence, the overall runtime is $O(1) + \sum_{i=1}^{n} O(i) = O(n^2)$.

Exercise 2: Bitstrings without consecutive ones *(12 Points)*

Given a positive integer n, we want to compute the number of n-digit bitstrings without consecutive ones (e.g., for $n = 3$ this number is 5, as 000, 001, 010, 100, 101 are the 3-digit bitstrings without consecutive ones).

(a) Give an algorithm which solves this problem in time $\mathcal{O}(n)$. Explain the runtime. *(7 Points)*

(b) Implement your solution. You may use the template `DP.py`. Run your algorithm on the values 10, 20 und 50 and write your results in `erfahrungen.txt`. *(5 Points)*

Sample Solution

(a) Let $A(n, i)$ be the number of n-digit bitstrings without consecutive ones ending on i, for $i \in \{0, 1\}$. We have $A(n, 0) = A(n-1, 0) + A(n-1, 1)$ and $A(n, 1) = A(n-1, 0)$. It follows $A(n, 0) = A(n-1, 0) + A(n-2, 0)$ and for the base cases $A(1, 0) = 1$ and $A(2, 0) = 3$. The recursive structure is the same as for the Fibonacci numbers. The calculation therefore goes along similar lines as in the lecture (week 11, slide 6). The number of n-digit bitstrings without consecutive ones is $A(n+1, 0)$.

(b) Cf. `DP.py` (another inplementation than described in (a)). The values for 10, 20 and 50 are 144, 17711 and 32951280099.