



# Algorithms and Datastructures

## Summer Term 2022

### Sample Solution Exercise Sheet 11

Due: January 18th, 12pm

#### Exercise 1: Bitstrings without consecutive ones (10 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. (5 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 implementation than described in (a)). The values for 10, 20 and 50 are 144, 17711 and 32951280099.

#### Exercise 2: Partitioning (10 Points)

Given a set  $X = \{x_0, \dots, x_{n-1}\}$  with  $x_i \in \mathbb{N}$ , we want to determine whether *there is* a subset  $S \subseteq X$  such that  $\sum_{x \in S} x = \sum_{x \in X \setminus S} x$  (it is not necessary to compute  $S$ ).

- (a) Let  $W := \sum_{x \in X} x$ . Give a recursive formula  $s : \{0, \dots, n-1\} \times \{0, \dots, W\} \rightarrow \{\text{True}, \text{False}\}$  such that  $s(i, j) = \text{True}$  if and only if there is a  $S \subseteq \{x_0, \dots, x_i\}$  such that  $\sum_{x \in S} x = j$ . Explain how  $s$  can be used to solve the above problem in time  $\mathcal{O}(W \cdot n)$ . (5 Points)
- (b) Implement your solution. You may use the template `DP.py`. Run your algorithm on the sets given in `set1.txt`, `set2.txt` and `set3.txt` and write your results to `erfahrungen.txt` (5 Points)

#### Sample Solution

- (a) Assume there is a set  $S \subseteq \{x_0, \dots, x_i\}$  with  $\sum_{x \in S} x = j$ . Then we either have  $x_i \in S$  or  $x_i \notin S$ . In the first case, there is a set  $S' \subseteq \{x_0, \dots, x_{i-1}\}$  with  $\sum_{x \in S'} x = j - x_i$ . In the second case there

is a set  $S'' \subseteq \{x_0, \dots, x_{i-1}\}$  with  $\sum_{x \in S''} x = j$ . If such a set  $S$  does not exist, there is neither  $S'$  nor  $S''$ . We therefore have that  $S$  exists if and only if  $S'$  or  $S''$  exist. We recursively define

$$s(i, j) = s(i - 1, j - x_i) \vee s(i - 1, j)$$

where  $\vee$  is the or-operator which is true if one of the arguments is true. We define the following base cases. We set  $s(i, 0) = \mathbf{True}$  since the empty set sums up to 0. We set  $s(0, j) = \mathbf{True}$  if and only if  $x_0 = j$ . We set  $s(i, j) = \mathbf{False}$  if  $i < 0$  or  $j < 0$ .

If there is a set  $S \subseteq X$  with  $\sum_{x \in S} x = \sum_{x \in X \setminus S} x$ , then both sums must equal  $W/2$ . We therefore obtain a solution of the problem by computing  $s(n - 1, W/2)$ .

We apply dynamic programming to compute  $s(n - 1, W/2)$ . As  $i$  and  $j$  only decrease in the recursion, we only have  $n \cdot (W/2 + 1) = \mathcal{O}(n \cdot W)$  different possibilities for parameters  $(i, j)$ . We therefore have to compute  $\mathcal{O}(n \cdot W)$  the value of  $s(i, j)$ .

Computing a single value via  $s(i, j) = s(i - 1, j - x_i) \vee s(i - 1, j)$  *without* the costs for the recursion takes  $\mathcal{O}(1)$ . We save all values  $s(i, j)$  in a dictionary `memo[i, j]` and therefore have to compute the value  $s(i, j)$  only once. As runtime we obtain  $\mathcal{O}(n \cdot W)$ .

- (b) Cf. `DP.py`. The results for the sets `set1.txt`, `set2.txt` and `set3.txt` are `True`, `False`, `True`.