University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Schneider

Algorithms and Data Structures
Winter Term 2019/2020
Exercise Sheet 11

Remark: This is a repetition exercise with a selection of previous topics based on your vote. Since next week will be the final lesson, there is no need to submit this exercise for feedback.

Exercise 1: Counting Bit Flips of a Binary Counter

Consider a counter represented as bit string. We increment (add 1 to) the counter n times. Show that the *amortized* number bit flips per increment operation is $\mathcal{O}(1)$. You may assume that your counter starts with 0 and has at least $\log_2 n$ bits.

(a) Analyze the number of bit flips using the *aggregate method.* That is, count the total number of bit flips and divide it by the number of operations.

(b) Analyze the number of bit flips using the *accounting method.* Specifically, show that by paying a constant amount of coins to an account per operation, and subtracting the true cost per operation from the account, the account still stays positive all the time.

Exercise 2: More Hashing

Let $h(s, j) := h_1(s) + j \cdot h_2(s) \bmod 13$ and let $h_1(x) = 2x + 3 \bmod 13$ and $h_2(x) = 2 + (x \bmod 12)$.

(a) Give an infinite key set (a subset of $\mathbb{N}$) that are mapped to the same table entry (for $j = 0$).

(b) Insert the keys **3,11,23,5,24,8,21,10** into the hash table of size $m = 11$ using the double hashing probing technique for collision resolution. The hash table below should show the final state.

0	1	2	3	4	5	6	7	8	9	10	11	12

Exercise 3: Frequent Numbers

You are given an Array $A[0 \dots n{-}1]$ of n integers and the goal is to determine frequent numbers which occur at least $n/3$ times in A. There can be at most three such numbers, if any exist at all.

(a) Give an algorithm with runtime $\mathcal{O}(n \log n)$ based on the divide and conquer principle that outputs the frequent numbers (if any exist).

(b) Argue why your algorithm is correct, give a recurrence relation for the runtime and use it to prove the runtime.

Exercise 4: Analysing an Algorithm

Algorithm 1 `algorithm`(A) ▷ **integer array** $A[0 \ldots n{-}1]$

> **for** $i \leftarrow 1$ **to** $n{-}1$ **do**
>   **for** $j \leftarrow 0$ **to** $i{-}1$ **do**
>   **for** $k \leftarrow 0$ **to** $n{-}1$ **do**
>   **if** $|A[i]-A[j]| = A[k]$ **then**
>   **return** true
> **return** false

(a) What does the above algorithm compute?

(b) Give the asymptotic running time of the above algorithm and a short explanation for that.

(c) Describe an algorithm that computes the same output but asymptotically faster.