Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Departement of Computer Science
Markus Püschel, David Steurer

29. October 2018

Algorithms & Data Structures — Homework 6 — HS 18

Exercise Class (Room & TA): _____

Submitted by: _____

Peer Feedback by: _____

Points: _____

Exercise 6.1 *iPhone Drop Test* ***(1 Point for Part 4)****.*

You just got a new job at Apple in the department of destructive testing. The first task given is to test the endurance of the new iPhone XR series. Specifically you need to determine the highest floor that the new iPhone can withstand when dropped out of the window.

When the phone is dropped and does not break, it is undamaged and can be dropped again. For simplicity assume that subsequent drops of the phone do not affect its endurance (i.e. if it survives it will have the identical state as if it weren't dropped at all). However, once the iPhone has been broken, you can no longer use it for another test.

If the phone breaks when dropped from floor n, then it would also have broken from any floor above that. If the phone survives a fall, then it will survive any fall below that.

As this is your first responsibility at your new job, you want to impress your new boss, and deliver results as soon as possible. To achieve that, you devise a strategy to minimize the number of drop tests required to find the solution.

1. What strategy would you use if only one phone is given and you perform the drop test on a building with n floors? What are the maximum number of drop tests that you have to perform?
2. What if we are given unlimited amount of identical phones?
3. What if we are given exactly 2 identical phones and the number of floors n is fixed such that $n = 100$?.
4. Assume that you are given 3 identical phones and a building with n floors. Determine the best search strategy for the floor where the phone breaks and give the number of drops in big-Θ notation. There is no need to prove that is best but only asymptotically optimal algorithms count.

Exercise 6.2 *Simple sorting.*

1. Perform two iterations of Bubble Sort on the following array. The array has already been partially sorted by previous iterations (after the double bar). By iterations we mean iterations of outer loop. You should only write two arrays corresponding to the end of first and second iterations.

9	5	8	13	15	10	11	7	6	20	21	35
1	2	3	4	5	6	7	8	9	10	11	12

2. Perform two iterations of Selection Sort on the following array. The array has already been partially sorted by previous iterations (up to the double bar). By iterations we mean iterations of outer loop. You should only write two arrays corresponding to the end of first and second iterations.

2	3	5	6	15	17	22	8	16	12	13	10
1	2	3	4	5	6	7	8	9	10	11	12

Exercise 6.3 *Inverse questions.*

1. Give a sequence of 5 numbers for which Bubble Sort performs exactly 10 swaps of keys in order to sort the sequence.
2. For all $n > 1$ give a sequence of n numbers for which Bubble Sort performs $\Theta(n\sqrt{n})$ swaps of keys in order to sort the sequence.
3. Assume that Selection Sort does not swap elements with the same index. For all $n > 1$ give a sequence of n numbers for which Selection Sort performs exactly 1 swap of keys in order to sort the sequence, but Bubble Sort and Insertion Sort perform at least $\Omega(n)$ swaps of keys.
4. For all $n > 1$ give a sequence of n numbers for which Bubble Sort, Selection Sort and Insertion Sort perform $\Theta(n)$ swaps of keys in order to sort the sequence.

Exercise 6.4 *Loop invariant **(1 Point)**.*

Consider the pseudocode of the MaxSubarraySum algorithm on an integer array $a[0, \ldots, n-1]$, $n \geq 1$.

```
procedure MaxSubarraySum(a)
    randmax ← 0
    max ← 0
    for 0 ≤ i < n do
        randmax ← randmax + a[i]
        if randmax > max then
            max ← randmax
        if randmax < 0 then
            randmax ← 0
    return max
```

Find a loop invariant INV such that:

1. INV(0) holds before the execution of the loop.
2. If INV(i) holds at the beginning of a loop iteration, then INV$(i+1)$ holds at the end of the loop iteration. Prove this.
3. INV(n) implies the correct solution.

Exercise 6.5 *TCP: Determine the maximum bandwidth* ***(1 Point)****.*

When transferring a large file over the internet, you want the file to arrive as fast as possible at the receiver. For this, the TCP protocol must determine the maximum bandwidth (e.g., measured in number of characters per second) which is available between sender and receiver. The available bandwidth is in general unlimited, time-dependent, and different for each transmitter-receiver pair. In this exercise, the task is to design a procedure (an algorithm) that is as efficient as possible to determine the available bandwidth.

For simplicity, we assume that during a connection the available bandwidth remains constant. The TCP protocol sends the data in each time unit with a bandwidth selected by the server. If the actual available bandwidth is sufficient (i.e., higher than the selected one), then the data arrives at the receiver. The receiver sends in this case indirectly before the end of the unit of time a confirmation back (the so-called Acknowledgement). If the bandwidth selected by the server has exceeded the available bandwidth, then the data sent in this time unit will be lost. The server detects this case by not receiving an acknowledgment from the receiver at the end of the time unit. So in each time unit one bandwidth can be tested by the server.

Design a procedure whereby the TCP protocol at the server determines the available bandwidth in as few time units as possible. What is the asymptotic number $O(f(b))$ of time units needed to compute the bandwidth b? (Assume that $b > 0$ is an integer.) Is your algorithm asymptotically optimal?

Submission: On Monday, 06.11.2018, hand in your solution to your TA *before* the exercise class starts.