Midterm Exam Data Structures and Algorithms 2017-2018

Friday September 29, 2017, 15.15–16.45

6 exercises

Explain your answer unless otherwise speficied!

Exercise 1. (*10 points*)
Are the following statements true or false? No motivation needed.

(a) The worst-case time complexity of heapsort is in $\Theta(n^2)$.

(b) The worst-case time complexity of insertion sort is in $\Theta(n^2)$.

(c) The time complexity of any sorting algorithm is in $\Omega(n \log n)$.

(d) The worst-case time complexity of quicksort is in $\Theta(n^2)$.

(e) The time complexity of insertion sort is in $\Theta(n^2)$.

Exercise 2. (*10 points*)
Give the worst-case time complexity in terms of Θ of the following sorting algorithms. No motivation needed.

(a) selection sort,

(b) bubble sort,

(c) quicksort,

(d) merge sort,

(e) counting sort.

Exercise 3. (*8+8 points*)
This exercise is concerned with merge sort.

(a) Give the recursion tree for the application of merge sort to the input array $[8, 4, 3, 5, 1, 2, 7, 6]$.

(b) Give a recurrence equation describing the time complexity of merge sort.

Explain your recurrence equation; connect with your recursion tree.

Solve your recurrence equation.

Exercise 4. (*6+8 points*)

(a) What is the lower bound on comparison-based sorting? No motivation needed.

(b) Give a decision tree for a comparison-based sorting algorithm for sorting an array of length 3 different natural numbers.

How many leaves should your tree have at least?

Exercise 5. (*8+10 points*)

(a) Consider the max-heap given by the array $[16, 14, 10, 8, 7, 9, 3, 2, 4, 1]$.

Depict the removal of 16 from the max-heap. Draw every step.

(b) Implement a stack using two queues; each queue has operations enqueue and dequeue in $\mathcal{O}(1)$.

Give (no motivation needed) the worst-case time complexity of your procedures pop and push in terms of $\mathcal{O}$.

Exercise 6. (*8+6+8 points*)
The following algorithm takes as input an array A consisting of n integers.

```
Algorithm Algo(A, n):
    max := 0
    for l := 1 to n do
        sum := 0
        for r := l to n do
            sum := sum + A[r]
            if sum > max then
                max := sum
    return max
```

(a) Apply Algo to the array $A = [-7, 8, -2, 3]$; consider all relevant l and r.

(b) What does Algo compute?

(c) Analyze the worst-case time complexity of Algo in terms of Θ.

The mark for the midterm is (the total number of points plus 10) divided by 10.