5. Problem set on

Algorithms, Data Structures and Data Abstraction WS 18/19

Wolfgang Mulzer

Due on 23. November 2018, 10:00, in the respective TA's mailbox

Problem 1 Specification and Implementation of an ADT *10 Points*

Give a meaningful verbal specification for the abstract data type "double ended queue" that allows insertion and deletion of elements on both ends of a queue. Try to be as exact as possible.

Write down a corresponding interface in Java, and implement the abstract data type in two different ways. Name respective advantages and disadvantages of your implementations (e.g., wrt running time, ease of implementation, flexibility, etc.).

(voluntary, 5 extra points) Proceed as in Problem 3.3(c) and create at least five JUnit-testcases for your implementation.

Problem 2 Lower Bounds for Priority Queues *5 Points*

(a) Show how to use a priority queue in order to sort a sequence of n pairwise comparable elements.

(b) In *AlP2*, you learned that every comparison-based sorting algorithm requires $\Omega(n \log n)$ comparisons. Conclude that in every comparison-based implementation of a priority queue, there is at least one operation with $\Omega(\log n)$ amortized running time.

Problem 3 Binary Heaps *15 Points*

(a) What is the minimum number of elements a binary heap with height h can have? What about the maximum number? Conclude that a binary heap with n elements must have height $\Theta(\log n)$.

Attention: We count the levels from 0.

(b) Where in a binary heap can the second smallest element be located? What about the third smallest element? Provide a general formula for the levels that may contain the k-smallest element in a binary heap with n elements. (You may assume that all levels are filled completely).

(c) Illustrate the bottom-up heap construction for `[7, 6, 5, 3, 1, 4, 2, 8]`.

(d) Suppose we insert n elements successively into a binary heap that is initially empty. Show that this takes $\Omega(n \log n)$ time in the worst case.

Attention: Initially, an insertion takes much less than $\log n$ steps.