Karl Bringmann, Erik Jan van Leeuwen **Winter 2016/2017**

Exercises for Algorithms and Data Structures

http://www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter16/algorithms-and-data-structures/

Exercise Sheet 11 Due: **30.1.2017**

The homework must be handed in on Monday before the lecture. You may collaborate with other students on finding the solutions for this problem set, but every student must hand in a writeup in their own words. We also expect you to state your collaborators and sources (books, papers, course notes, web pages, etc.) that you used to arrive at your solutions.

You need to collect at least 50% of all points on all exercise sheets to be admitted to the final exam.

Whenever you are asked to design an algorithm in this exercise sheet you have to give a proof of its correctness as well as an asymptotic upper bound on its worst case running time.

Exercise 1 (*10 points*)

Consider an implementation of a FIFO queue data structure using an array of size N. If the queue expands to more than N items, then the array is re-allocated to an array of size $2N$, and all items are copied from the old array to the new array. Show that the amortized cost of each queue operation in a sequence of m operations is $O(1)$.

Exercise 2 (*10 points*)

Implement a FIFO queue data structure using two stacks, such that amortized cost of each enqueue and dequeue operation in a sequence of m operations is $O(1)$. You must use the potential method.

Exercise 3 (*10 points*)

a) (3 points) Describe and analyze a `union` algorithm that applies union-by-rank, but when both sides have equal rank, it makes the side that has the lowest value the root of the joint tree.

b) (7 points) Consider the disjoint sets data structure implementation by union-find, where we use the algorithm of a) to perform `union`, and we use path compression in `find`. Consider the following sequence of operations:

1. `make-set`(1)
   $\vdots$
8. `make-set`(8)
9. `union`$(1, 2)$
10. `union`$(3, 4)$
11. `union`$(4, 5)$
12. `union`$(6, 7)$
13. `union`$(6, 8)$
14. `union`$(4, 7)$
15. `find`(4)
16. `find`(7)

Show the answers returned by the `find`-operations, and show the contents of the data structure (trees, ranks, values) both after line 14 and after line 16.

Exercise 4 (*10 points*)

Consider the disjoint sets data structure implementation by union-find, where we use union-by-rank and the normal find (that is, without path compression).

a) (3 points) If we use the normal find (that is, without path compression), is it true that $\text{rank}(\text{parent}(x)) = \text{rank}(x) + 1$ for any node x that has a parent? Is this also true if we use path compression? In both cases, prove your answer is correct or give a counterexample.

b) (4 points) Show that the amortized cost of `make-set` is $O(1)$, and of `find` and `union` are $O(\log n)$.

   Hint: Bound the rank.

c) (3 points) Give a sequence of operations which achieves these bounds.