58131 Data Structures
V exercise, week 48/2003, English translation

Exercise V.1: Lecture slides 6.3.4 explained, how priority queues can be implemented on top of red-black trees. (This material is not in the book, sorry.) Write the corresponding *complete* pseudocode including

- all the required tree operations, even their left/right symmetric branches
- the heap operation calls to be added to these tree operations
- for all heap operations required, versions that operate on trees instead of arrays
- all priority queue operations using this data structure, even handles (for handles, see page 139 of the book).

Exercise V.2: Consider different kinds of hashing methods. The hash table size is $m = 11$ and the following keys are added in the following order:

$$10, 22, 31, 4, 15, 28, 17, 88, 59$$

Explain, how the table progressively fills up, when each of the the following methods is used:

Collision resolution by chaining when the hash function is $h(x) = x \bmod m$.

Linear probing when $h' = h$.

Quadratic probing when $h' = h$, $c_1 = 1$ and $c_2 = 3$.

Double hashing when $h_1 = h$ when $h_2(x) = 1 + (x \bmod (m-1))$.

Exercise V.3: Consider open addressing, when a full table is doubled when necessary. (Doubling is explained in Chapter 17.4 of the book.) Use quadratic probing as follows: The hash function uses the mutiplication method with Knuth's suggestion, as explained in Chapter 11.3.2 of the book (and lecture slides 7.2). The probe sequence is the one in Problem 11.3 of the book (and the first suggestion on lecture slide 287).

The initial size of the table is $m = 4$. Key additions are as in Exercise V.2. Explain, how the table progressively fills up, overflows, and expands.

Exercise V.4: The node $s \in V(G)$ of a directed graph G is a *universal sink* if there is an edge $r \to s \in E(G)$ into s from every node $r \in V(G) \setminus \{s\}$ of G except perhaps s itself.

Let G be given as an adjacency matrix. Write an algorithm, which determines in $\mathcal{O}(|V(G)|)$ steps whether or not G has such a universal sink.

Exercise V.5: You are planning a flight trip from Helsinki to Kabul. You fear flying, and want to choose the safest possible route, even if it is not the shortest.

As background information you have

- a list of all commercial airports in the worlds and all direct regular flights between them (you must use only them, you cannot for instance hire a private plane)
- for every such flight

- the number of passengers who have taken this flight up to now
- how many of these passengers met with an airplane accident during their flight.

Explain, how you can still turn this problem into one of finding a shortest path in some weighted graph. (This course will provide algorithms for this latter problem in the future.)

Exercise V.6: These were the final exercises of this course. Give therefore (in addition to your normal group evaluations) suggestions how this course and its exercises should be developed in the future. Discuss your ideas with other groups as well!

(In addition to this, you personally can give anonymous feedback electronically at the end of the course, as usual.)

(Total number of exercises: 6 pcs.)