Practice Exam for

Algorithmen und Programmierung III

Wolfgang Mulzer WiSe 2013/14

Time: 90 minutes.

Aufgabe 1 Implementations of an ADT *2+2+3+3*

Consider the following specification of an abstract data type: Let $\mathcal{U}$ be a totally ordered universe. We would like to store subsets $S \subseteq U$, so that the following operations are possible:

- `insert`(x): Pre: None. Effect: $S \mapsto S \cup \{x\}$.
- `deleteMin`(): Pre: S is not empty. Effect: $S \mapsto S \setminus \{\min S\}$.
- `deleteMax`(): Pre:: S is not empty. Effect: $S \mapsto S \setminus \{\max S\}$.

You may assume that two elements from $\mathcal{U}$ can be compared in constant time. For each of the following data structures, describe briefly how to implement the operations `deleteMin` and `deleteMax` efficiently, and give good asymptotic upper bounds on the running times. If necessary, explain what additional assumptions need to be made.

(a) Sorted doubly linked list with pointers to the first and last element;

(b) AVL-tree;

(c) binary min-heap; and

(d) uncompressed trie.

Aufgabe 2 Huffman-Codes *1+6+3*

(a) When is a code $C : \Sigma \to \{0,1\}^*$ prefix-free? What makes this definition useful?

(b) Let $\Sigma = \{a, b, c, d, e, f\}$ with frequencies a : 12, b : 62, c : 80, d : 74, e : 25 and f : 86. Use Huffman's algorithm to construct a code for Σ and the given frequencies. Show the individual steps.

(c) Evaluate the following statement: "Huffman-codes can be used to compress a given file optimally."

Aufgabe 3 Miscellaneous *3+2+2+3*

(a) What is a design pattern? What is an algorithm? Give one difference and one common property.

(b) What is the difference between the static and the dynamic data type of a variable? Give a short example.

(c) Under which circumstances is a data structure with $O(\log n)$ *amortized* time per operation preferable to a data structure with $O(\log n)$ *worst-case* time per operation?

(d) In class, you have seen several algorithms that use dynamic programming. Why do these algorithms first find the *value* of an optimal solution, before finding an optimal solution itself?

Aufgabe 4 Skip-Lists *4+3+3*

(a) Show that the expected size of a skip-list with n elements is $O(n)$.

(b) Let L_1 and L_2 be skip-lists, with element sets K_1 and K_2, respectively. Give an efficient algorithm to obtain a skip-list for the element set $K_1 \cup K_2$ from L_1 and L_2. Analyze the expected running time of your algorithm. (You may use the result from (a) without proof.)

(c) Suppose that in (b) we also know that for all $k_1 \in K_1$, $k_2 \in K_2$ we have $k_1 < k_2$. This means that every element in L_1 comes before every element in K_2. Describe an algorithm that merges L_1 and L_2 in expected time $O(\max\{\log |K_1|, \log |K_2|\})$. Prove the guarantee on the running time.

Hint: For $x \in (0,1)$, we have $\sum_{i=1}^{\infty} ix^i = x/(1-x)^2$.