Aalto University School of Science and Technology
Department of Computer Science and Engineering
T-106.1227 Data Structures and Algorithms

EXAMINATION
27.10.2011

Write on each paper your name, student number, degree programme, and the course code with name. Also write the date, hall, the number of papers you return, and your *signature*.

1) Ten Questions (10 x 1p)

This is a *compulsory part* of the final exam. You need to get at least 5p out of the maximum 10p so that the rest of the exam will be checked. However, this part alone is not enough to pass the whole exam. On the other hand, in order to get 5p, you are not required to give "the exactly correct answer", but more or less show that *you have understood the functionality of the code fragments* related to this part. Thus, pay attention to the reasoning. Refer to the code line numbers if possible.

In the following, you can see two algorithms (`traverse1` and `traverse2`) for traversing a binary tree. Read through all the questions below without answering them and after that familiarize yourself with the code throughout. After this, answer all the questions and take time for pondering and explaining your reasoning. Note, however, that all the questions refer to the given algorithms. In addition, the claims in the questions can be justified to be either true or false, thus the *argumentation* is the only thing that matters for the points.

```
1 void traverse1(node t) {
2    if (t != NULL) {
3       visit(t);
4       traverse1(t->left);
5       traverse1(t->right);
6    }
7 }
8
9
10
```

```
11 void traverse2(node root) {
12    stack.push(root);
13    while (stack.notEmpty()) {
14       node next = stack.pop();
15       visit(next);
16       if (next->right != NULL)
17          stack.push(next->right);
18       if (next->left != NULL)
19          stack.push(next->left);
20    }
21 }
```

a) *Describe* how algorithm `traverse1` works in general (without an example). Note! Try to answer *how* the given algorithm solves the computational problem – do not just explain the code line-by-line.

b) *Describe* how algorithm `traverse2` works in general. How this is different compared with the previous one?

c) *Give an example* of `traverse1` in case of Fig. 1. starting from node P. Hint: Show in tabular form which nodes the variables `t`, `t->left`, and `t->right` refer during the execution of the algorithm. What is the output of the algorithm if `visit(t)` prints the letter of the visited node?

d) *Give an example* of `traverse2` in case of Fig. 1 starting from node P. In this time, tabulate variables `next`, `next->left`, and `next->right`. What is the printing order in this case?

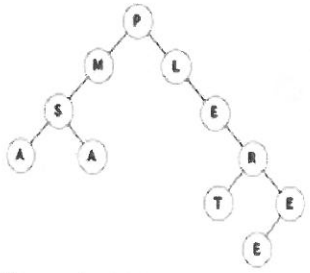


Figure 1: A binary tree

e) *Analyze* the time complexity of `traverse1` if the number of nodes in the tree is n.
f) *Analyze* the time complexity of `traverse2` if the number of nodes in the tree is n. Remember to justify your analysis (the end result is not enough for the points).
g) *Argue* whether it is true or false: `traverse1` is more efficient than `traverse2`.
h) *Argue* whether it is true or false: `traverse1` does more comparisons (if-statements) than `traverse2`.
i) *What* would be the traversing order in case you swap lines 3 and 4 in `traverse1`? *Give an example*.
j) Algorithm `traverse2` utilizes a *while*-loop. Is it possible to replace this loop with some other loop structure? Either argue why this is not possible or write a new algorithm to show how.
Bonus exercise:
k) *Ponder and compare* the memory consumption of `traverse1` and `traverse2`.

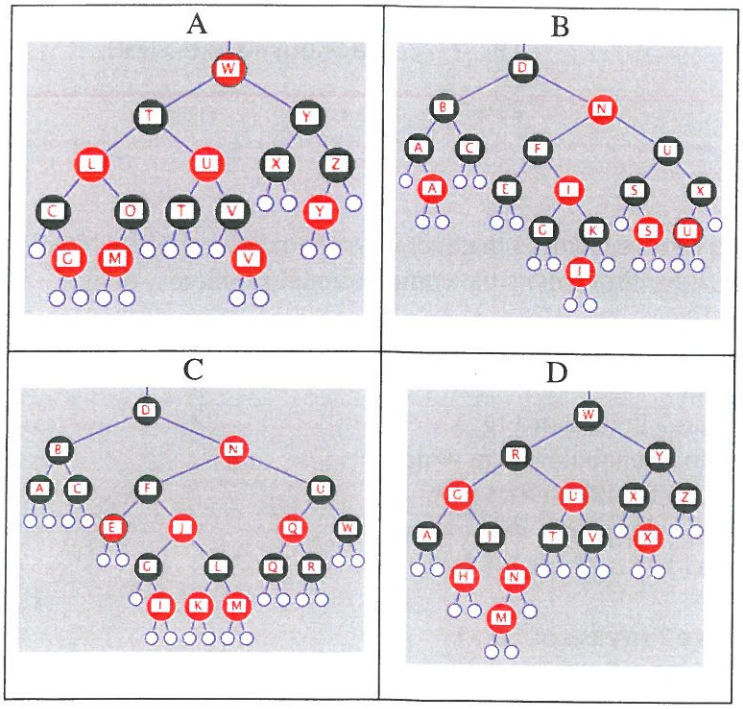
2) Terminology (2p + 2p + 2p + 2p)

Define the following *concepts* (4 x 1p). In addition, *give an example* of each (4 x 1p).

a) Heap (keko)
b) Hash function (hajautusfunktio)
c) Quadratic probing (neliöllinen kokeilu)
d) Selection problem (valikointi-ongelma)

3) Red-Black Tree (4p + 4p)

a) *Define* Red-Black Tree.
b) Which of the figures depict valid Red-Black Trees, and which rules in your definition the rest of the cases do not satisfy?



4) Sorting methods (10p)

Write an essay on sorting methods. Start from orientation to the topic, and among other things, define what you mean by sorting. After this, proceed towards the details. You can discuss broadly different kinds of sorting methods by comparing them with each other. Or, you can limit your essay to a certain efficient sorting method by describing it in detail and giving examples. The length of the essay should be about two pages.

5) Graphs (4p + 2p + 4p)

a) *Describe* the recursive *Depth-First Search algorithm* (DFS). Assume the input is an undirected graph in adjacency list representation, and each adjacency list is ordered alphabetically.
b) *Give the adjacency list representation* for the graph in Figure 2.
c) *Show* how the recursive DFS works on the graph of Figure 2 in case the starting node is A. Show the discovery (visiting order) and finishing times (i.e., when recursive calls return) for each vertex.

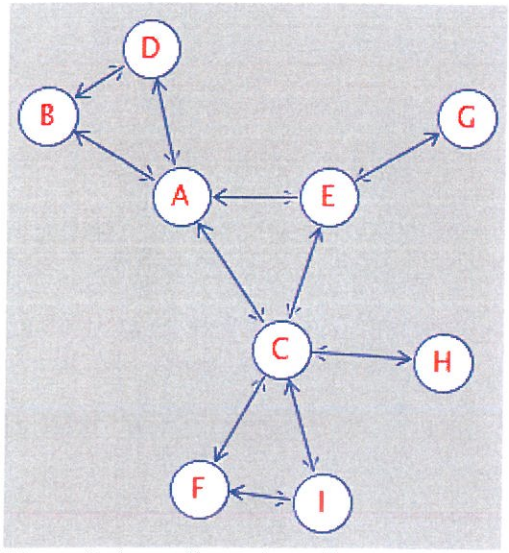


Figure 2: An undirected graph.