Aalto University School of Science
Department of Computer Science
CS-A1143 Data Structures and Algorithms Y

EXAMINATION
17.12.2019

Write on each paper your name, student number, degree programme, and the course code and name. Also write the date, hall, the number of papers you return, and your *signature*. It is forbidden to use any extra equipments in this examination.

1) Ten Questions (10 x 1p + 1p)

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

In the following, you can see two algorithms computing the factorial. Read through all the questions below without answering them and only 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 based on code comprehension* is the only thing that matters for the points!

```
1 def fact_1(n):
2   if (n < 2):
3     return 1
4   else:
5     return fact_1(n - 1) * n
```

```
 6 def fact_2(n):
 7     fact = 1
 8     for i in range(1, n + 1):
 9         fact = fact * i
10     return fact
```

a) *Describe in one sentence* how `fact_1` computes factorials *without* any example.
b) *Describe in one sentence* how `fact_2` computes factorials *without* any example. Make sure your descriptions differ and point out the main difference between the two algorithms.
c) *In which order and how many* multiplications `fact_1` does in line 5? Give *an example* in case fact_1(4).
d) *In which order and how many* multiplications `fact_2` does in line 9? Give *an example* in case fact_2(4).
e) *Analyze* the time complexity of `fact_1` in terms of the input size n.
f) *Analyze* the time complexity of `fact_2` in terms of the input size n.
g) *Argue* whether it is true or false: `fact_1` is more efficient than `fact_2`.
h) *Argue* whether it is true or false: `fact_1` computes the same function than `fact_2`.
i) *What is the order of multiplications* in `fact_1` if line 5 would be changed to `return n * fact_1(n - 1) * n`? *Give an example.*
j) Is it possible to replace the for-loop in `fact_2` with another loop? Argue either why not or give an example how to replace it (write the algorithm anew).

Bonus exercise:
k) *Ponder and compare* the memory consumption of `fact_1` and `fact_2`.

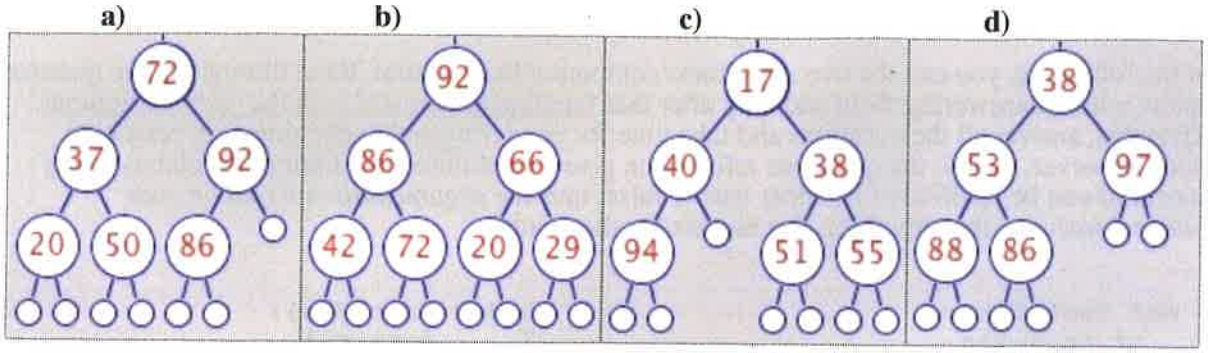
2) Terminology (2p + 2p + 2p + 2p)

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

a) *Stable* sorting method
b) *In place* sorting
c) *Inversion*
d) *Selection problem*

3) Priority Queues (2p + 2p + 2p + 2p)

Consider the following four binary trees in Figures a, b, c, and d. *For each case, argue* whether the figure depicts a MinHeap (yes/no, 1p) and *justify your answer* (1p).



4) Balanced Search Trees (2p + 6p + 4p)

a) *Define* the concept Balanced Search Tree.
b) What kind of balanced search trees you know? How do they differ from each other? *Deal with three different data structures.*
c) *Give an example* (*draw and mark* (give a name) the intermediate states before and after *each balancing operation*) of a balanced search tree in case the items
`11,5,2,18,7,4,3,6,10,5,6,5` will be inserted into the tree in this order. You can assume the duplicates will be inserted into the right branch of the tree.

5) Graph algorithms (7p + 3p)

a) *Write an algorithm* to find the connected sub-graphs of a given undirected graph.

b) *Analyze* the running time of your algorithm.

You can represent the algorithm by using some known programming language or pseudo-code. However, explain the working of the algorithm also verbally.