Aalto University School of Science
Department of Computer Science
CS-A1141 Data Structures and Algorithms

EXAMINATION
12.12.2016

Write on each paper your name, student number *with the control letter*, 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)

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 computing the factorial. 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 int fact_1(int n)
2 {
3    if (n<2)
4        return 1;
5    else
6        return fact_1(n-1)*n;
7 }
```

```
 8 int fact_2(int n)
 9 {
10    int i,fact;
11    fact = 1;
12    for (i=1; i<=n; i++)
13        fact = fact*i;
14    return fact;
```

a) *Describe* how Algorithm 1 (i.e., fact_1) works by using an appropriate *example*.
b) *Describe* how Algorithm 2 (i.e., fact_2) works by using an appropriate *example*.
c) *In which order and how many* multiplications fact_1 does? *Give an example* in case the algorithm is called with parameter n=3.
d) *In which order and how many* multiplications fact_2 does? *Give an example* in case the algorithm is called with parameter n=3.
e) *Analyse* the time complexity of Algorithm 1 in terms of the input size n.
f) *Analyse* the time complexity of Algorithm 2 in terms of the input size n.
g) *Argue* whether it is true or false: Algorithm 1 is more efficient than Algorithm 2.
h) *Argue* whether it is true or false: Algorithm 1 computes the same function than Algorithm 2.
i) *What is the order of multiplications* in Algorithm 1 if the line 6 would be changed to "return n*fact_1(n-1);"? *Give an example*.
j) Is it possible to replace the for-loop in Algorithm 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 Algorithm 1 and 2.

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

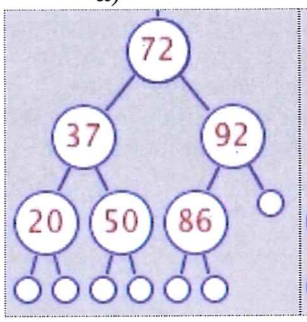
Define the following *concepts* (4 x 1p). In addition, *give an example* of each (4 x 1p).

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

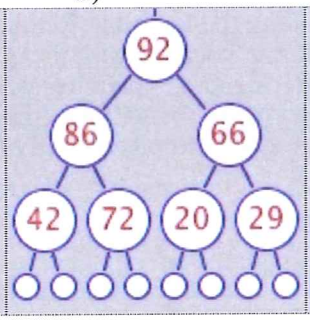
3) Priority Queues (2p + 2p + 2p + 2p)

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).

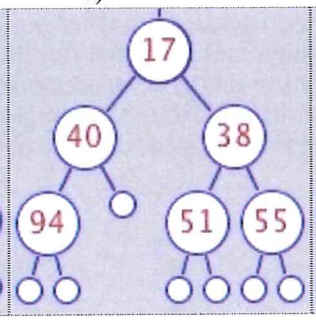
a)



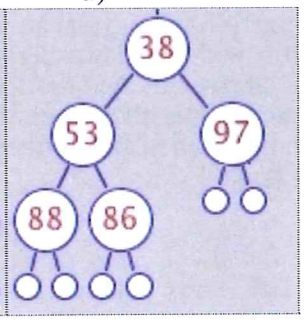
b)



c)



d)



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.

Aalto-yliopiston perustieteiden korkeakoulu
Tietotekniikan laitos
CS-A1141 Tietorakenteet ja algoritmit Y

TENTTI
12.12.2016

Kirjoita jokaiseen paperiin oma nimesi, oppilasnumerosi *tarkistusmerkkeineen*, tutkinto- tai koulutusohjelmasi, kurssikoodi ja kurssin nimi, päivämäärä, sali, palauttamiesi paperien lukumäärä sekä *allekirjoituksesi*. Tentissä ei saa käyttää mitään ylimääräisiä apuvälineitä.

1) Kymmenen kysymystä (10 x 1p)

Tämä tehtävä on *tentin pakollinen osa*, josta on saatava vähintään 5p/10p, jotta loput tentistä tarkistetaan. Tämä tehtävä ei kuitenkaan yksistään riitä tentin läpäisyyn. Toisaalta viiteen pisteeseen ei edellytetä "täysin oikeaa vastausta" vaan oleellista on, että pystyt osoittamaan *ymmärtäneesi* tehtävän *koodin* toiminnan. Käytä siis aikaa perustelujen miettimiseen ja esittämiseen. Viittaa perusteluissa ohjelmakoodin rivinumeroihin, jos mahdollista.

Alla on annettu kaksi algoritmia (fact_1 ja fact_2), jotka laskevat kertoma-funktion (*factorial*). Lue ensin kaikki kysymyskohdat vastaamatta niihin ja sen jälkeen tutustu annettuihin koodinpätkiin erittäin huolella. Vastaa tämän jälkeen kaikkiin kysymyksiin ja käytä aikaa perustelujen pohtimiseen ja muotoilemiseen. Huomaa, että kaikissa kysymyksissä viitataan alla oleviin algoritmeihin ja, että väittämät voi perustella yhtä hyvin vastaamalla kyllä tai ei, joten pisteet tulevat vain *perusteluista*!

```
1 int fact_1(int n)
2 {
3    if (n<2)
4        return 1;
5    else
6        return fact_1(n-1)*n;
7 }
```

```
 8 int fact_2(int n)
 9 {
10    int i,fact;
11    fact = 1;
12    for (i=1; i<=n; i++)
13        fact = fact*i;
14    return fact;
15 }
```

a) *Selitä* algoritmin fact_1 toiminta *esimerkin avulla*.
b) *Selitä* algoritmin fact_2 toiminta *esimerkin avulla*.
c) *Missä järjestyksessä ja kuinka monta* kertolaskua fact_1 suorittaa? *Anna esimerkki*, kun algoritmia kutsutaan parametrillä n=3.
d) *Missä järjestyksessä ja kuinka monta* kertolaskua fact_2 suorittaa? *Anna esimerkki*, kun algoritmia kutsutaan parametrillä n=3.
e) *Analysoi* algoritmin 1 suoritusaika sen saaman syötteen koon n funktiona.
f) *Analysoi* algoritmin 2 suoritusaika sen saaman syötteen koon n funktiona.
g) *Perustele* pitääkö väite paikkansa vai ei: algoritmi 1 on tehokkaampi kuin algoritmi 2.
h) *Perustele* pitääkö väite paikkansa vai ei: algoritmi 1 laskee saman funktion kuin algoritmi 2.
i) *Mikä* olisi algoritmin 1 *kertolaskujen suoritusjärjestys*, jos riviä 6 muutettaisiin muotoon "return n*fact_1(n-1);"? *Anna esimerkki*.
j) Algoritmi 2 käyttää *for*-silmukkaa. Voitaisiinko se korvata jollakin toisella silmukalla? Perustele joko miksi ei tai anna esimerkki miten (kirjoita algoritmi uusiksi).

Bonustehtävä:
k) *Pohdi ja vertaile* algoritmien 1 ja 2 muistinkäyttöä.

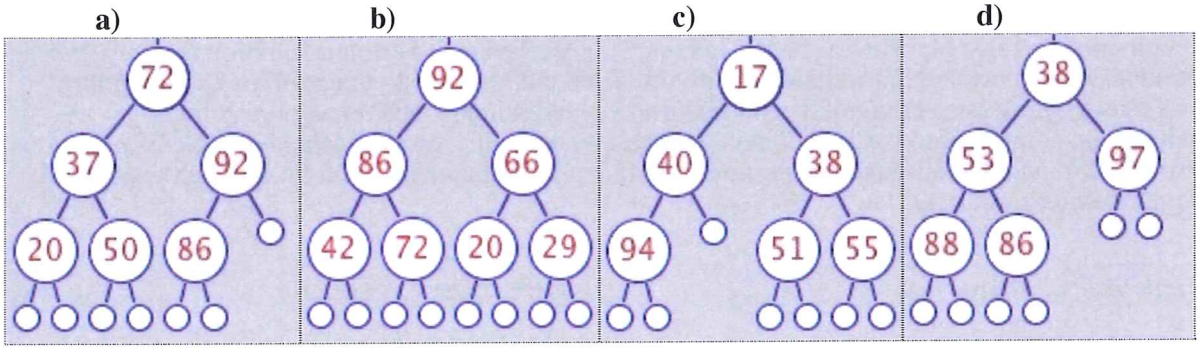
2) Terminologiaa (2p + 2p + 2p + 2p)

Määrittele seuraavat *käsitteet* (4 x 1p). *Anna* jokaisesta myös *esimerkki* (4 x 1p).

a) Stabiili (*stable*) järjestämismenetelmä
b) Paikoillaan tapahtuva (*in place*) järjestäminen
c) Inversio (*inversion*)
d) Valikointi (*selection problem*)

3) Prioriteettijonot (2p + 2p + 2p + 2p)

Seuraavissa kuvissa on neljä binääripuuta (*binary tree*) a, b, c ja d. *Vastaa jokaisen kohdalla erikseen* onko kuvan tietorakenne *minimikeko* (*MinHeap*) (kyllä/ei, 1p) ja *perustele* miksi (1p).



4) Tasapainotetut hakupuut (2p + 6p + 4p)

a) *Määrittele* käsite *tasapainotettu hakupuu* (*balanced search tree*).
b) Millaisia erilaisia tasapainotettuja hakupuita on olemassa? Miten ne eroavat toisistaan? *Käsittele* tässä *kolmea eri tietorakennetta*.
c) *Anna esimerkki* (*piirrä ja merkitse* (nimeä) välivaiheet ennen ja jälkeen *jokaisen tasapainotusoperaation*) jonkin tasapainotetun hakupuun toiminnasta, kun siihen lisätään alkiot `11,5,2,18,7,4,3,6,10,5,6,5` tässä järjestyksessä. Samanarvoiset alkiot sijoitetaan aina oikeanpuoleiseen haaraan.

5) Verkkoalgoritmit (7p + 3p)

a) *Kirjoita algoritmi*, joka etsii annetun suuntaamattoman verkon yhtenäiset osaverkot (*connected sub-graphs*).
b) *Analysoi* algoritmisi suoritusaika.

Voit esittää algoritmin jollakin tunnetulla ohjelmointikielellä tai käyttää vapaampaa pseudokieliesitystä. Selitä algoritmin toiminta kuitenkin myös sanallisesti.