Aalto-yliopiston perustieteiden korkeakoulu
Tietotekniikan laitos
CSE-A1140/[1141] Tietorakenteet ja algoritmit [Y]

TENTTI
24.5.2016

Kirjoita jokaiseen paperiin oma nimi, oppilasnumero, tutkinto-ohjelma, kurssikoodi ja kurssin nimi, päivämäärä, sali, palauttamiesi paperien lukumäärä sekä *allekirjoituksesi*. Numeroi palauttamasi paperit juoksevalla numeroinnilla. Tentissä ei saa käyttää mitään ylimääräisiä apuvälineitä.

1) Kymmenen kysymystä (10 x 1p + 1p = 11p)

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 (`bs1` ja `bs2`), jotka molemmat etsivät puolitushaulla järjestetystä taulukosta `table` alkiota `x`. 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 vastaukset tulee perustella hyvin, tai siis *pisteet tulevat vain perusteluista*!

```
1  int bs1(int table[], int x) {
2    int low = 0;
3    int high = table.length - 1;
4    int mid;
5
6    while( low <= high )
7    {
8      mid = (low + high) / 2;
9
10     if (table[mid] < x)
11       low = mid + 1;
12     else if (table[mid] > x)
13       high = mid - 1;
14     else return mid;
15   }
16   return -1;
17 }
```

```
18 def bs2(table, x, low, high):
19   if low > high:
20     return -1
21   mid = (low + high) / 2
22   item = table[mid]
23   if item == x:
24     return mid
25   elif x < item:
26     return bs2(table,x,low,mid-1)
27   else:
28     return bs2(table,x,mid+1,high)
29
30
31
32
33
```

a) *Selitä* algoritmin `bs1` toiminta sanallisesti (ilman esimerkkiä). Huom! Pyri selittämään *miten* algoritmi ratkaisee ongelman. Älä selitä koodia rivi-riviltä.
b) *Selitä* nyt algoritmin `bs2` toiminta sanallisesti. Miten toiminta eroaa edellisestä?
c) *Anna esimerkki hausta*, jossa algoritmilla `bs1` etsitään alkiota `x` = 512 taulukosta, jossa on alkiot 1, 2, 4, 8, 16, 32, 64, 128, 256 ja 512. Vihje: Taulukoi mitä arvoja muuttujat `low`, `high` ja `mid` saavat ohjelman suorituksen edetessä. Mitä ohjelma palauttaa ja mitä laskennan tuloksena saadaan?
d) *Anna esimerkki hausta*, jossa algoritmilla `bs2` etsitään em. taulukosta arvoa `x` = 100. Vihje: taulukoi tässä muuttujat `low`, `high`, `mid` ja `item` kuten edellä. Mitä ohjelma palauttaa ja mikä on laskennan tulos tässä tapauksessa?
e) Määrittele `bs1`:n ja `bs2`:n ns. "syötteen koko", ts. mistä muuttujista ja miten algoritmien suoritusajat riippuvat?
f) *Analysoi* `bs1`:n suoritusaika sen saaman syötteen koon *n* funktiona.
g) *Analysoi* `bs2`:n suoritusaika sen saaman syötteen koon *n* funktiona.
h) Algoritmia `bs1` testattiin suurella aineistolla (haettiin aineiston pienintä alkiota), jolloin sen suoritusajaksi saatiin noin 1 millisekunti. Tämän jälkeen aineiston koko kaksinkertaistettiin ja suoritusajaksi saatiin noin 2 millisekuntia. Jos aineisto jälleen kaksinkertaistettaisiin, niin kuinka pitkään arvioisit laskennan tällä kertaa kestävän? Perustele.
i) Millaisia oletuksia ja reunaehtoja `bs2`:n virheetön ja tehokas suoritus asettaa taulukolle `table` ja taulukon sisältämille alkioille?
j) *Perustele* pitääkö väite paikkansa vai ei: `bs1` on tehokkaampi kuin `bs2`.
k) Bonustehtävä: *Pohdi ja vertaile* `bs1`:n ja `bs2`:n muistinkäyttöä.

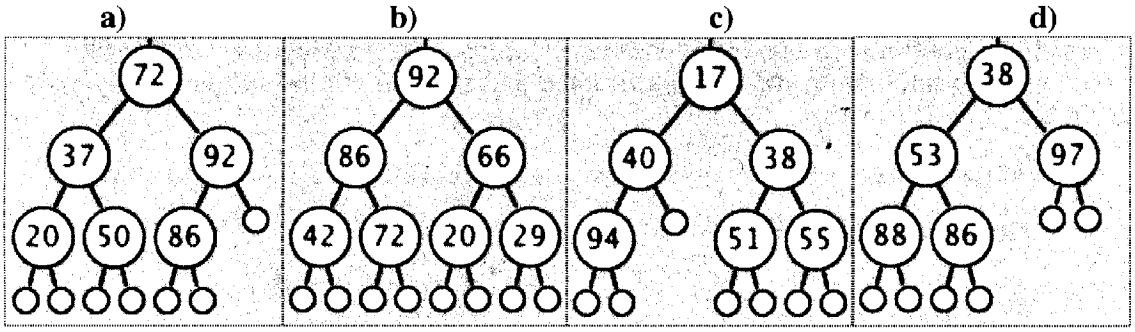
2) Terminologiaa (2p + 2p + 2p + 2p + 2p)

Määrittele seuraavat *käsitteet* (4 x 1p). Huom! *Anna* jokaisesta myös *esimerkki* (4 x 1p). Yritä määritellä käsite mahdollisimman yleisesti ja anna sen jälkeen sen jostakin yksittäistapauksesta esimerkki nimeämällä jokin erikoistapaus, piirtämällä, tms.

a) Abstrakti tietotyyppi (ADT, Abstract Data Type)
b) Asymptoottinen suoritusaika (asymptotic running time)
c) Hajautusfunktio (Hash function)
d) Vierusmatriisi (Adjacency Matrix)
e) Vahvasti yhtenäiset komponentit (Strongly Connected Components)

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) Ajankäyttö (0p)

Arvioi tenttiin vastaamiseen käyttämäsi aika 15 minuutin tarkkuudella.