
3 tehtävää, yhteensä 40p. Kokeessa ei saa käyttää mitään oheismateriaalia.

1. [Yhteensä 20p] Vastaa *lyhyesti* seuraaviin kysymyksiin:

- Mitä tarkoitetaan (täsmällisesti!) algoritmin iso- O aika- ja tilavaativuudella. (5p)
- Miksi haluamme hakupuiden olevan tasapainoisia? Mitkä ehdot hakupuun on täytettävä jotta se olisi AVL-puu? (5p)
- Piirrä jokin sellainen binäärihakupuu (ei välttämättä tasapainoinen, eikä välttämättä AVL-puu) jossa on vähintään viisi solmua, ja juurella on sekä vasen että oikea lapsi. Piirrä solmuihin myös avaimet, jotka ovat kokonaislukuja. Tee puun juurisolmulle vasen kierto (ja piirrä lopputulos). (5p)
- Miten käsittelet törmäykset hajautuksessa? Kerro kaksi eri vaihtoehtoa, ja kerro (lyhyesti) niiden hyvät ja huonot puolet. (5p)

2. [Yhteensä 10p] Halutaan toteuttaa linkitettyinä listana tietotyyppi, johon kohdistuu kolmenlaisia operaatioita:

- $\text{Search}(S, k)$: avaimen k haku joukosta S (kuten tavallisella joukolla)
- $\text{Insert}(S, x)$: alkion x lisäys joukkoon S (kuten tavallisella joukolla)
- $\text{Split}(S, k, P, Q)$: Luodaan uudet joukot P ja Q . Joukkoon P tulee kaikki ne joukon S alkiot, joiden avain on korkeintaan k . Loput alkiot tulevat joukkoon Q . Operaatiossa joukko S tulee tyhjäksi.

Aiotusta sovelluksesta tiedetään, että yleisimpiä operaatioita ovat alkion lisääminen ja joukossa olevan avaimen hakeminen; Split on harvinaisempi

- Minkä muunnelman linkitetystä listasta valitsisit? Perustele lyhyesti. Mitkä ovat Insert- ja Search-operaatioiden aikavaativuudet valitsemassasi rakenteessa? (3p)
- Esitä Split-operaation toteutus pseudokoodina käyttäen valitsemasi tyyppisiä listoja. Mikä on aika-vaativuus? (4p)
- Olisiko tietotyyppi mielestäsi kuitenkin parempi toteuttaa tasapainoisilla hakupuilla? Esittele lyhyesti kummankin ratkaisutavan hyviä ja huonoja puolia. (Älä rupea tarkasti miettimään Split-operaation hakupuutoteutusta. Vetoaminen tietorakenteiden tunnettuihin yleisiin ominaisuuksiin riittää.) (3p)

3. [Yhteensä 10p] **Huom:** Tee vain toinen kohdista (a) tai (b), jos teet molemmat, vain (a) kohta arvostellaan.

- Kirjoita algoritmi $\text{HAKUPUU}(x)$, joka saa parametrikseen binääripuun x , ja palauttaa True, jos puu on hakupuu, ja False jos se ei ole hakupuu. Algoritmin aikavaativuuden tulee olla $O(n)$, kun puussa on n solmua. Kuvaa ongelman ratkaisu sanallisesti, ja anna pseudokoodi. Perustele miksi ratkaisusi toimii ajassa $O(n)$. (Hitaammastakin (mutta toimivasta) algoritmista saa osapisteitä.)
- Oletetaan, että AVL-hakupuun jokaiseen solmuun x on liitetty kenttä $size$ (siis $x.size$), joka kertoo solmun x jälkeläisten lukumäärän, mukaan lukien solmu x itse.

Kirjoita algoritmi $\text{SELECT}(x, i)$, joka palauttaa (ali)puun x solmuista sen, jonka avain on i :nneksi suurin. Algoritmin tulee toimia ajassa $O(\log n)$, kun puussa on n solmua (avainta). Kuvaa ongelman ratkaisu sanallisesti, ja anna pseudokoodi. Perustele miksi ratkaisusi toimii ajassa $O(\log n)$.
