Tietorakenteet ja algoritmit 1 **Palautus 6.5.2010 klo 16:00**

Harjoitus 8

2 kpl X-tehtäviä

X-tehtävien ratkaisujen pitää olla kunkin opiskelijan itse tekemiä. Saman ratkaisun kopioita ei hyväksytä(versioitunakaan). Vastaukset pitää lähettää **torstaina 6.5.2010 klo 16.00** mennessä sähköpostitse alla olevaa ohjetta käyttäen. Saat automaattisen vastauksen pian onnistuneen lähetyksen jälkeen. Vastauksen on sisällettävä lyhyt itsearviointi jossa arvioit ratkaisun toimivuutta, aikavaativuutta ja mahdollisia parannusmahdollisuuksia. Oikea itsearvointi (jonkinlaiseen ratkaisuun) on yhden pisteen arvoinen.

Lähetä ratkaisusi cs:n käyttäjälle mhk käyttäen viestin otsikkona merkkijonoa TRA1_X3_omakayttajatunnus X3-tehtävässä ja TRA1_X4_omakayttajatunnus X4-tehtävässä, missä omakayttajatunnus on sinun cs-käyttäjätunnuksesi. Helpointa lähettäminen on cs:ltä käyttäen komentoa:

/usr/ucb/mail -s TRA1_X3_omakayttajatunnus mhk < omakayttajatunnus.java

/usr/ucb/mail -s TRA1_X4_omakayttajatunnus mhk < omakayttajatunnus.java

missä omakayttajatunnus on cs-käyttäjätunnuksesi ja omakayttajatunnus.java on ohjelmatiedosto joka sisältää vastauksesi. Jotta tehtävä kääntyisi, on pääohjelman luokan nimen oltava täsmälleen sama kuin käyttäjätunnuksesi (kuitenkin ilman @cs tms. osuutta). Ohjelman on siis oltava sähköpostin runkona sellaisenaan (ilman MIME/HTML -koodauksia, allekirjoituksia, uusia rivityksiä tms.). Ratkaisun pisteytys lasketaan suoraan mukaan kurssin pisteisiin.

X3) Toteutetaan binääripuun läpikäynti sisäjärjestyksessä takaperin. Tähän tarvitaan (a) algoritmi joka hakee binääripuun äärimmäisenä oikealla sijaitsevan solmun (solmun jolla ei ole oikeaa lasta) ja (b) algoritmi joka hakee binääripuun solmun edeltäjän sisäjärjestyksessä. Kirjoita siis algoritmit inorderLast() ja inorderPrevious(). Solmun edeltäjä on joko vasemman lapsen oikeanpuoleisin jälkeläinen, tai, jollei vasenta lasta ole, niin se esivanhempi jonka oikeassa alipuussa alkuperäinen solmu oli. Huomaa, ettei puun alkioiden tarvitse olla sisäjärjestyksessä. Itseasiassa et edes tarvitse alkioita lainkaan, puun rakenne riittää. Ota pohja www-sivulta.

X4) Toteutetaan sanakirja (operaatiot insert, contains, remove, sekä läpikäynti) suljetulla hajautuksella. Alkiotyyppinä on geneerinen tyyppi ja hajautukseen käytetään ko. tyypin hashCode() metodia. Varsinaisena hajautustauluna toimii taulukko (ei Vector tms.). Luokan runko on www-sivulla. Täydennä runkoon operaatiot insert(E x) ja contains(E x):

insert() lisää alkion sanakirjaan jollei sitä siellä jo ennestään ole. Talletusaluetta ei tarvitse täytteyessä laajentaa, poikkeus riittää.

contains() palauttaa totuusarvon onko alkio kokoelmassa vai ei.

Näidenkin operaatioiden on huomioitava mahdollisuus, että sanakirjasta on poistettu alkioita, eli täydennä ratkaisuasia operaatiolla remove(E x) joka poistaa annetun alkion sanakirjasta. Poistossa alkio korvataan erikoisarvolla removed jotta myöhemmät haut onnistuisivat.