Tietorakenteet ja algoritmit II 7.11.2019

Harjoitus 2

Piirrä kuva kustakin tehtävästä ja suunnittele algoritmisi toiminta kuvan kanssa ennen toteuttamisen aloittamista. Kaikissa "kirjoita algoritmi joka" tehtävissä tehdään toimiva Java-metodi joka saa parametrinaan syötteen ja palauttaa tehtävänannon mukaisen palautusarvon, mutta ei tee mitään muuta. Ei siis esimerkiksi muuta syötteitään ellei syötteen muuttamista erikseen pyydetä. Lopullinen versio itse algoritmimetodista ei myöskään tulosta mitään. Ota syötteen generoiva ja metodia kutsuva pääohjelma Moodlesta. Koska algoritmitehtävät on tarkoitus tehdä toimiviksi asti, tuo toimivat ohjelmasi testattavaksi harjoitustilaisuuteen USB-tikulla, verkkoon tallennettuna tai omalla tietokoneella esitettäväksi.

Seuraava tehtävä X2 on pakollinen kaikille opiskelijoille. X-tehtävien ratkaisujen pitää olla kunkin opiskelijan **itse** tekemiä. Saman ratkaisun kopioita ei hyväksytä (versioitunakaan). Vastaukset pitää lähettää keskiviikkona 6.11. klo 21:00 mennessä allaolevaa ohjetta käyttäen. Saat automaattisen vastauksen pian onnistuneen lähetyksen jälkeen. Jollet saa kuittausta, on lähetyksessä mennyt jotain väärin. Jos kuittauksessa on kääntäjän virheilmoituksia, on tiedostossa tai sen lähetyksessä jotain väärin. Lähetä tällöin **korjattuna uudestaan**. Vastauksen on sisällettävä ohjelman kommenteissa lyhyt **itsearviointi** jossa arvioit ratkaisun toimivuutta, aikavaativuutta ja mahdollisia parannusmahdollisuuksia. Oikea itsearviointi (jonkinlaiseen ratkaisuun) on yhden pisteen arvoinen.

Lähetä ratkaisusi www-lomakkeella jonka osoitteen ja tunnukset olet saanut erillisessä sähköpostissa. Vastauksen tulee olla kääntyvä Java-ohjelma nimeltään `TRAI_X2_`*`tunnus`*`.java`, missä *`tunnus`* on alkuosa siitä sähköpostiosoitteesta jonka olet tälle kurssille Oodissa antanut. Alkuosa päättyy ensimmäiseen pisteeseen (.), tavuviivaan (-) tai @-merkkiin. *`tunnus`* kaikki pienillä kirjaimilla. Esim. `sjuva@uef.fin` → `TRAI_X2_sjuva.java` ja `simo.juvaste@uef.fin` → `TRAI_X2_simo.java`. Jotta ohjelma kääntyisi, on myös luokan nimen oltava täsmälleen sama kuin tiedoston nimi, huomioiden isot ja pienet kirjaimet. Luokan on myös toteutettava rajapinta `TRAI_X2` jonka löydät Moodlesta. Rajapintamäärittelyä tai testi/pääohjelmaa ei pidä sisällyttää mukaan lähetykseen, ne ovat eri tiedostossa.

Ota runko vastaukseen Moodlesta äläkä muuta varsinaisen tehtävämetodin otsikkoa (nimeä, parametreja, tyyppejä). Älä myöskään tee luokastasi pakettia. Moodlessa on myös testiohjelma jota käyttämällä ja muokkaamalla voit testata oman ratkaisusi toimintaa.

X2. Kirjoita metodi joka mittaa listan lisäys- ja poisto-operaatioiden nopeutta. Kullekin syötteen koolle n mitataan miten kauan menee lisätä aluksi tyhjään listaan n alkiota ja poistaa ne sieltä. Lisäykset ja poistot tapahtuvat listan loppuun/lopusta. Tulos palautetaan nanosekunteina. Metodi saa parametrinaan tyhjän listan (*java.util.List*) sekä kaksi kokonaislukua *min* ja *max* ja mittaa aikavaativuuden syötteen koolle *min* ja kaksinkertaistaen syötteen koon aina *max* asti. Tuloksena metodi palauttaa kuvauksen (*TreeMap*<*Integer*, *Long*>) jossa löytyy tulokset mitatuille syötteen koille. Vaadittu ratkaisu on kaksiosainen, ensimmäinen osa mittaa ajankäytön yhdelle syötteen koolle (n) ja toinen osa kutsuu ensimmäistä osaa pyydetyille arvoille ja muodostaa tuloksista kuvauksen. Tehtävän arvostelussa otetaan huomioon paitsi se, että on mitattu oikeaa asiaa, myös se miten hyvin mittaus minimoi häiriötekijät. Hyvään arvosanaan sinun on siis suoritettava mittaus useasti. Kerro itsearvioinnissa miten minimoit virhelähteet ja mitä mahdollisia epävarmuuksia mittaukseesi vielä jäi.

5. Kirjoita ohjelma jolla voit vertailla *HashSet*:n ja *TreeSet*:n toiminnan nopeutta samoin kuin tehtävällä X2 pystyi vertailemaan *LinkedList*:in ja *ArrayList*:n nopeutta. Hyödynnä tehtävän X2 ratkaisua ja testiohjelmaa.

6. Kirjoita ohjelma joka testaa operaation *java.util.TreeSet.subSet*() aikavaativuuden. Mikä näyttäisi olevan aikavaativuuden kertaluokka? Miten tulos on selitettävissä? Onko aikavaativuus optimaalinen?

7. Kirjoita ohjelma joka testaa operaatioiden *java.util.HashMap.containsKey*() ja *java.util.HashMap.containsValue*() aikavaativuuden. Mitkä näyttäisivät olevan aikavaativuuksien kertaluokat? Miten tulokset on selitettävissä?

8. Todista seuraavat väittämät määritelmään nojautuen todeksi tai epätodeksi (johda epäyhtälöt siten, että ylhäältä/alhaalta rajoittaminen tulee näytetyksi ja c ja n_0 selviävät):

   (a) $4n^2-2n+1=\Omega(n^2)$

   (b) $n^2-2n+4=\Omega(n^2)$

   (c) $n\sqrt{n}-n+2=o(n^2)$

   (d) $n^2-4n+4=\Theta(n^2)$