Tietorakenteet ja algoritmit I

26.11.2019

Kertaustehtäviä

1. Aseta seuraavat lausekkeet suuruusjärjestykseen.

1000×2	1000^2	$2^{50} - 1000$	$1000^2 + 5$
1000×5	1000×50	$1000/5$	$1000 - 5$
$1000^2/5$	$1000 + 5$	$1000^2 \times 5$	$1000^2 + 50$

2. Aseta seuraavat funktiot kasvunopeuden mukaiseen järjestykseen.

$n\sqrt{n}$	$n\log n$	$(\log n)^2$	$n/5$
$n\log\log n$	n^2	$(4/3)^n$	$n^2 + \log n$
$n^2/\log n$	n^4	$n/\log n$	$n^2 \log n$

3. Anna seuraavien operaatioiden aikavaativuuksien kertaluokat kun n on kyseisessä kokoelmassa olevien alkioiden määrä ja m parametrikokoelman alkioiden määrä (niissä kohdissa joissa parametrina on kokoelma). Huomioi myös kunkin kohdan parametrin tyyppi.
   Vihje: kunkin operaation aikavaativuus on joko $O(1)$, $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n \log m)$ $O(m \log n)$, $O(nm)$ tai $O(n^2)$. Aikavaativuudet pystyy päättelemään kunkin kokoelman toteutustavasta. Muista, ettet saa käyttää ohjelmoidessasi kokoelmia/operaatioita joita et tunne.

   (a) *ArrayList.get*(*int*)

   (b) *LinkedList.remove*(*int*)

   (c) *TreeMap.put*(*Object, Object*)

   (d) *HashSet.retainAll*(*HashSet*)

   (e) *ArrayList.contains*(*Object*)

   (f) *LinkedList.add*(*Object*)

Seuraavissa "kirjoita algoritmi" -tehtävissä on tarkoitus kirjoittaa täsmällistä Java:n tapaista algoritminotaatiota kuten olemme kurssilla käyttäneet. Täsmällistä Javan syntaksia tai operaatioiden nimiä ei vaadita, mutta annettuja/valittuja tietorakenteita on käytettävä oikein. Esimerkiksi binääripuulla ei ole lisäys- tai hakuoperaatiota, mutta puun solmulla on operaatio jolla voi asettaa vasemmaksi lapseksi jonkin toisen solmun. Jos olet epävarma jonkin operaation nimestä tai parametreista, käytä selkeää toimintaa kuvaavaa nimeä ja/tai erillistä selitystä. Alkiot ovat samat jos niiden *.equals*() -metodi antaa toden.

Muista tehdä huolellinen suunnitelma ennen algoritmin kirjoittamisen aloittamista. Ilman kuvallista suunnitelmaa algoritminkirjoitus ei suju.

4. Kirjoita algoritmi joka poistaa annetusta listasta L kaikki ne alkiot jotka esiintyvät listassa L vain yhden kerran. Listaan siis jäävät ne alkiot jotka esiintyvät vähintään kaksi kertaa. Älä muuta jäljellejääneiden alkioiden järjestystä. Käytä listana valintasi mukaan joko *java.util.LinkedList*, *java.util.ArrayList*, tai *TraLinkedList* listaa ja kerro valintasi. Vihje: apurakenteita saa käyttää. Mikä on algoritmisi aikavaativuus? Pyri tehokkaaseen algoritmiin.

5. Kirjoita algoritmi joka lisää sisäjärjestettyyn binääripuuhun alkion siten, että binääripuu säilyy sisäjärjestyksessä. Jos sama alkio on jo binääripuussa, alkiota ei kuitenkaan lisätä. Syötteenä ovat siis binääripuu ja alkio, palautusarvona totuusarvo *true* jos lisäys tehtiin, muuten muuten *false*. Mikä on algoritmisi aikavaativuus?