Tentamen i Algoritmer & Datastrukturer i Java

Hjälpmedel: Skrivhjälpmedel, miniräknare.
Ort / Datum: Halmstad / 2008-08-18
Skrivtid: 4 timmar
Kontakt person: Nicolina Månsson tel. 035-167487
Poäng / Betyg: Max 44 poäng
_ >=18 poäng ger betyg 3
_ >=27 poäng ger betyg 4
_ >=36 poäng ger betyg 5

Övrigt: Det finns två olika sorters frågekategorier, de som skall besvaras genom programmerings och de som skall besvaras genom förklaringar med text och illustrationer.
Programmeringslösningarna skall i största mån vara skrivna i korrekt Java-syntax.
Koden skall vara välstrukturerad och lättläst.
Koden skall innehålla lämpliga ledtexter (utskrifter) som instruerar användaren vad som krävs av honom / henne.
Om en lösning är uppenbart ”klumpig” och det anses att tentanden skall känna till en smidigare lösning kan den ”klumpiga” lösningen medföra poängavdrag.
Förklaringslösningar bör, om tillämpningsbart, innehålla illustrationer på relevanta datastrukturer och använda algoritmer.
Tänk på att vara noggrann och strukturerad. Det är Du som skall visa vad Du kan!

Lycka Till!

Uppgift 1 – Tidskomplexitet (4p+2p+2p)

a) För nedanstående delar av olika algoritmer ange en uppskattning av exekveringstiden i form av Big-Oh notation. Motivera också din uppskattning.

a1)
```
for( int i=1; i<2n; i++)
    x=x+1;
```

a2)
```
for( int i=1;i<n*n; i++)
    x=x+1;
for(int j=1;j<n; j++)
    x=x+3;
```

a3)
```
for(int i=n;i>0;i=i/2)
    x=x+1;
}
```

a4)
```
i=n;
while(i>=1){
 for( int j=1;j<n;j++)
        x=x+1;
 i=i/2;
}
```

b) När två program A och B analyseras uppskattas deras exekveringstid till 150NlogN respektive N^2.
Vilket program har i verkligheten den bästa exekveringstiden för små värde av N (N<100)?
Vilket program har bäst exekveringstid för stora värde (N>10 000)? Motivera svaret!
c) En viss algoritm exekveras i 0,5 ms för en input storlek 100. Vad blir exekveringstiden om inputstorleken ändars till 500, då algoritmen uppskattas som O(N^2).
Ledning! Exekveringstiden T(n)=k*F(n) där F(n) är just tillväxtfunktionen som matchar algoritmen.

Uppgift 2 – Datastrukturer (2p+1p+2p+4p)

a) Om ett dataprogram ska simulera något av nedanstående situationer, vilka datastrukturer är mest lämpade? Motivera svaret.

a1- Flygplan som väntar i luften i väntan på att landa
a2- En lexikon, där den vanligaste operationen är sökning
a3- En alfabetiskt lista med namn, där nya namn ska in och ut.
a4 - Exekveringen av en rekursiv metod

b) Förklara kort fördelar och nackdelar mellan de två olika implementationerna av en kö, med array och med länkad lista.

c) I kursboken ges det programkod för en sk. *”wrap-around”* implementation av en kö.
Förklara datastrukturen kö genom att använda ett enkelt exempel.

d) Om du hade behövt skriva ut innehållet i en kö borde klassen kompletteras med en metod printQueue(). Beskriv med ord eller exempel en lösning för den nämnda metoden.

Uppgift 3 – Kända algoritmer (4p+2p+2p)

a) Anta att vi måste söka efter ett element bland en stor mängd data som är lagrad i en array. Vad måste vi veta om datan som skall sökas igenom för att kunna tillämpa binärt sökning?
Binary search algoritmen kan implementeras både iterativt och rekursivt. Implementera en lösning för algoritmen. Algoritmen skall fungera för arrayer av heltal, se nedan.

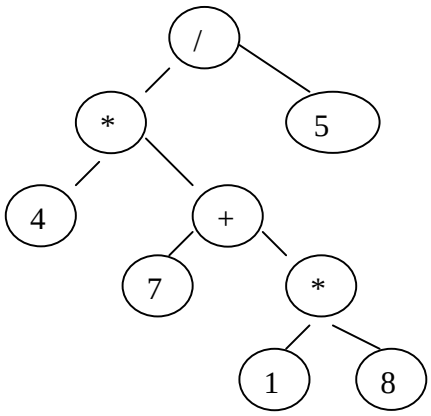
```
public int binarySearch (int [] a, int x){

}
```

b) I vanligt fall är *quicksort* algoritmen en mycket effektiv sorterings algoritm men *insertionsSort* kan vara att föredra i visa speciella fall.
I vilken/vilka fall är insertionSort algoritmen ett bättre val då du skall sortera en viss mängd data?

c) Förklara varför *quicksort* algoritmen kan ge exekveringstid i O(N^2) i worst-case och O(NlogN) i best-case.

Uppgift 4 – Träd och Huffman **(2p+2p+4p+2p)**

Vi kan använda binära träd för att representera algebriska uttryck. Dessa träd kallas ”expressions tree”.
För följande träd:



a) Om man skriver ut innehållet i trädet genom att använda metoden printPostOrder(), kommer du att uppnå en aritmetisk uttryck i *postfix notation*. Vilken blir uttrycket för ovanstående träd.

b) För att beräkna aritmetiska uttryck i postfix notation används också datastrukturen **operator stack**. Vissa hur stacken förändras när du beräknar *postfix uttrycket* som du fått fram i uppgift a.

c) En fil innehåller: mellanslag(605), semikolon(705), 0(431), 1(242), 2(176), 4(185), 5(250),7(199), 8(205), 9(217) ska komprimeras med Huffman algoritmen. När jag bygger Huffman trädet hittar jag följande koder för mina tecken.

(se nästa sida)

- ; får 11
- får 010
- 0 får 001
- 5 får 0000
- 1 får 0001
- 9 får 1000
- 8 får 1001
- 7 får 1010
- 4 får 1011
- 2 får 01110

Din uppgift är att rita upp trädet som motsvarar koderna och beräkna hur mycket filen har komprimerats då ursprungligen varje tecken behövde 8 bitars minne.

d) Implementera en rekursiv metod som beräknar antalet löv i ett binärt träd. Metoden har följande signatur:

static int leaves(BinaryNode n){ }

Uppgift 6 – Länkad lista (3p+6p)

Din uppgift är att implementera en enkel länkad lista av heltal. Den består av två klasser, klassen som definierar en Node och själva klassen LinkedList som länkar noderna.

a) Implementera klasen Node.
b) Implementera klassen LinkedList med konstruerare och enbart 2 metoder : insert(int tal) och remove(int tal)