Linköpings Tekniska Högskola
Institutionen för Datavetenskap
Tommy Färnqvist

11 januari 2012

TDDC70/TDDC91 Datastrukturer och algoritmer
Tentamen 2012-01-11, 08–13 (TER1)

Examinator:	Tommy Färnqvist
Jour:	Tommy Färnqvist (telefon 070 4547668).
Max poäng:	28 poäng (betyg 5 = 24p, 4 = 19p, 3 = 14p)
Hjälpmedel:	INGA HJÄLPMEDEL TILLÅTNA!!!

VÄNLIGEN IAKTTAG FÖLJANDE

- Lösningar till olika problem skall placeras enkelsidigt på separata blad. Skriv inte två lösningar på samma papper.
- Sortera lösningarna innan de lämnas in.
- MOTIVERA DINA SVAR ORDENTLIGT: avsaknad av, eller otillräckliga, förklaringar resulterar i poängavdrag. Även felaktiga svar kan ge poäng om de är korrekt motiverade.
- Om ett problem medger flera olika lösningar, t.ex. algoritmer med olika tidskomplexitet, ger endast optimala lösningar maximalt antal poäng.
- SE TILL ATT DINA LÖSNINGAR/SVAR ÄR LÄSBARA.
- Lämna plats för kommentarer.

Lycka till!

1. Omöjligt? Din kompis Egbert är känd för att komma med fantastiska påståenden. Vilka av Egberts påståenden nedan är möjliga respektive omöjliga? Svar utan motivering ger inga poäng. (3 p)

   (a) Egbert säger: Jag har en ny algoritm som kan sortera listor i linjär tid, förutsatt att varje elements position i den osorterade listan inte är mer än 1000 steg bort från rätt position i den sorterade listan. (1)

   (b) Egbert säger: Jag har en ny algoritm som kan hitta det största talet i en given osorterad array i logaritmisk tid. (1)

   (c) Egbert säger: Jag har en ny algoritm som kan sortera vilken lista som helst i linjär tid, bara jag får en lämplig jämförelsefunktion. (1)

2. Om `a` är en array av längd m, där varje element `a[i]` är ett heltal mellan 0 och $n-1$, vad är värstafallstiden för en körning av följande kodsnutt? (3 p)

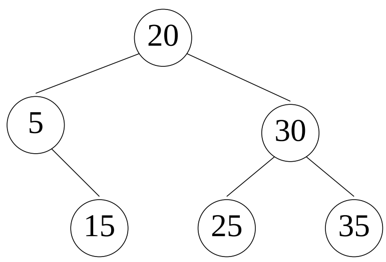
```
k = 0;

for ( int i = 0; i < m; ++i ) {
   for ( int j = 0; j < a[i]; ++j ) {
      b[k] = i;
      ++k;
   }
}
```

   Om det också är känt att summan av alla element i `a` är n, vad blir då exekveringstiden för kodsnutten ovan?

3. Träd (7 p)

   (a) Den *totala längden* av ett träd T är summan av avstånden från roten av T till alla andra noder i T. Beskriv en algoritm som får som indata roten r till ett binärt träd och som returnerar den totala längden av trädet. Analysera tidskomplexiteten hos din algoritm. (3)

   (b) Betrakta följande AVL-träd. Sätt in nyckeln 10 i trädet och balansera om det vid behov. (1)



   (c) En grupp biologer lagrar information om DNA-strukturer i ett AVL-träd där de använder strukturens specifika vikt (ett heltal) som nyckel. Biologerna behöver hela tiden svara på frågor av typen "Finns det några strukturer i trädet med specifik vikt mellan a och b (inklusivt)?" och de hoppas på så snabba svar som möjligt. (3)

   Beskriv en algoritm som, givet heltal a och b, returnerar sant om det finns en nyckel x i trädet, sådan att $a \leq x \leq b$, och falskt om ingen sådan nyckel existerar i trädet. Vad har din algoritm för tidskomplexitet?

4. Vi använder en array med indexen 0 till 6 för att implementera en öppet adresserad hashtabell av längd 7 med $h(k) = k \bmod 7$ som hashfunktion. Rita en representation av hashtabellen och dess innehåll efter insättning av nycklarna 7, 21, 42, 10 och 22 i tabellen: (4 p)

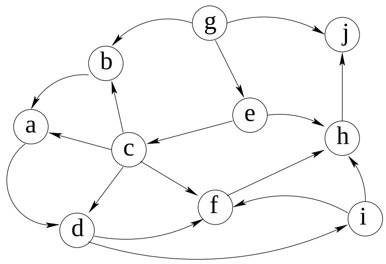
   (a) när linjär sondering (*linear probing*) används för att hantera kollisioner (2)

   (b) när dubbel hashning, med $h'(k) = 5 - (k \bmod 5)$ som sekundär hashfunktion, används för att hantera kollisioner. (2)

5. En array är *bitonisk* om den innehåller en strikt ökande sekvens av nycklar omedelbart följd av en strikt avtagande sekvens av nycklar. Beskriv en algoritm som hittar den största nyckeln i en bitonisk array av längd N i tid proportionell mot $\log(N)$. (4 p)

6. Grafer (7 p)

   Betrakta följande riktad graf G där alfabetisk ordning över nodnamn bestämmer ordningen på grannarna till varje nod.



   (a) Visa i vilken ordning noderna i G besöks av en breddenförstsökning med start i a. (1)

   (b) Visa i vilken ordning noderna i G besöks av en djupetförstsökning med start i a. (1)

   (c) Beräkna en topologisk ordning för G. (2)

   (d) Skriv om följande påstående som en sats om oriktade grafer och bevisa satsen. Antag att om A är vän till B, så är B vän till A och att för varje A gäller att A inte är vän till A. "I varje grupp av $n \geq 2$ personer finns det två personer med samma antal vänner i gruppen." (3)