Danmarks Tekniske Universitet

Skriftlig prøve, den 26. maj 2009.

Kursusnavn Algoritmik og datastrukturer I Kursus nr. 02105.

Tilladte hjælpemidler: Alle skriftlige hjælpemidler.

Vægtning af opgaverne: Opgave 1 - 24%, Opgave 2 - 16%, Opgave 3 - 23%, Opgave 4 - 12 %, Opgave 5 - 25 %.

Vægtningen er kun en cirka vægtning.

Alle opgaver besvares ved at udfylde de indrettede felter nedenfor. Som opgavebesvarelse afleveres blot denne og de efterfølgende sider i udfyldt stand. Hvis der opstår pladsmangel kan man eventuelt benyttes ekstra papir som så vedlægges opgavebesvarelsen.

Opgave 1 (kompleksitet)

1.1 Angiv for hver af nedenstående udsagn om de er korrekte:

	Ja	*Nej*
$n^7 = O(n^3)$	☐	☐
$n(\log n)^3 = O(n^2)$	☐	☐
$2^n = O(n^2)$	☐	☐
$n^2 + \frac{1}{2}n = \Omega(n)$	☐	☐
$n(n-3)/17 = \Theta(n^2)$	☐	☐

1.2 Skriv følgende liste af funktioner op i voksende rækkefølge efter asymptotisk vækst. Dvs. hvis funktionen $g(n)$ følger umiddelbart efter funktionen $f(n)$ i din liste, så skal der gælde at $f(n) = O(g(n))$.

$4\sqrt{n}$

$2/\log n$

$\frac{1}{2}n^3$

$\frac{2}{3}n$

$(\log n)^5$

Svar: _____

1.3 Antag at du har en algoritme hvis køretid er præcist $5n^2$. Hvor meget langsommere kører algoritmen hvis du forbdobler inputstørrelsen?

A dobbelt så langsom B 4 gange langsommere C 5 gange langsommere

D 10 gange langsommere E 20 gange langsommere

1.4 Betragt nedenstående algoritme.

```
Algoritme Løkke1(n)
1.   x = 1
2.   for i = 1 to n
3.      for j = 1 to n
4.         for k = 1 to n
5.            x = x + 1
```

a) Køretiden af algoritmen er

A $\Theta(\log n)$ B $\Theta(n)$ C $\Theta(n \log n)$ D $\Theta(n^2 \log n)$ E $\Theta(n^3)$

F $\Theta(n^{3/2})$ G $\Theta(2^n)$ H $\Theta(n^4)$ I $\Theta(\sqrt{n})$

b) Hvis linie 4 ændres til "**for** $k = j$ **to** n" så bliver køretiden:

A asymptotisk langsommere

B asymptotisk den samme

C asymptotisk hurtigere

1.5 Betragt nedenstående algoritme.

Algoritme Løkke2(n)
1. $i = 1$
2. **while** $i \leq n$ **do**
3. $j = 1$
4. **while** $j \leq n$ **do**
5. $j = j + 1$
6. $i = 2i$

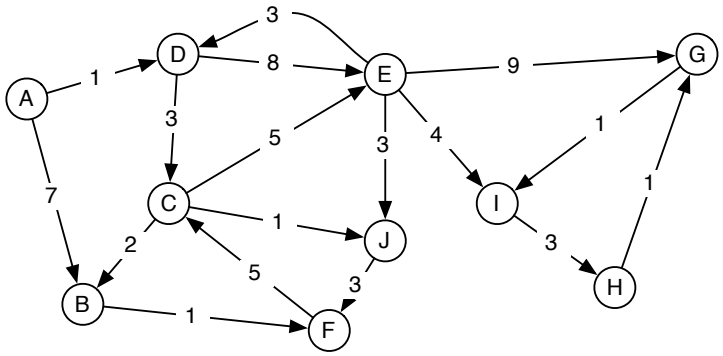
Køretiden af algoritmen er

A $\Theta(\log n)$ B $\Theta(n)$ C $\Theta(n \log n)$ D $\Theta(n^2 \log n)$ E $\Theta(n^3)$

F $\Theta(n^{3/2})$ G $\Theta(2^n)$ H $\Theta(n^4)$ I $\Theta(\sqrt{n})$

Opgave 2 (grafer)

2.1 Betragt nedenstående graf G med ikke-negative kantvægte. Det antages at incidenslisterne er sorteret alfabetisk.

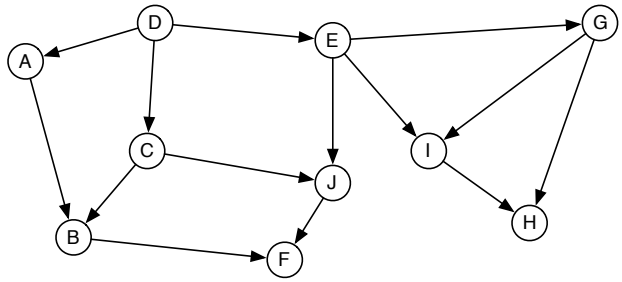


a) Angiv et BFS træ for grafen G når BFS genemløbet starter i knuden A. Angiv BFS-dybde/lag for hver knude. Det antages at incidenslisterne er sorteret i alfabetisk orden.

b) Angiv et DFS træ for grafen G, når DFS genemløbet starter i knuden A. Angiv en DFS nummerering af knuderne (en DFS nummerering er den rækkefølge knuder bliver besøgt i). Det antages at incidenslisterne er sorteret i alfabetisk orden.

c) Angiv et korteste veje træ for grafen G når korteste veje beregningen sker med hensyn til startknuden A. Angiv for hver knude afstanden fra knuden A.

2.2 Betragt nedenstående DAG D.



Hvilke af følgende rækkefølger angiver en topologisk sortering af knuderne i grafen D:

1 D E A C B J F I G H	2 D A E C B I J G H	3 D A B C E J F G I H
4 D E A C B J F G I H	5 D A E C B J G H	6 D E A C F B J G I H

Opgave 3 (modellering, anvendelse og analyse af algoritmer)

Du er konsulent for langtursbusfirmaet "Blåhundebusserne" der ønsker at lave en hjemmeside hvor folk kan finde og bestille deres rejse. Man skal både kunne søge på den billigste rejse mellem to valgfrie destinationer og på den rejse med færrest antal omstigninger (busskift). Der er S forskellige stoppesteder/stationer og B forskellige busruter. For hver busrute kender du startsted og destination, samt prisen for at benytte ruten. En bus stopper ikke undervejs, men kører direkte fra startsted til destination. Du skal ikke tage hensyn til tidspunkt for afgang og ankomst i denne opgave.

3.1 Giv en effektiv algoritme der finder den billligste rejse mellem to givne destinationer s og t. Rejsen kan være en kombination af flere forskellige busruter. Prisen for rejsen bliver så summen af priserne for de ruter der benyttes. Angiv køretiden af din algoritme i asymptotisk notation. Din analyse skal være så tæt som mulig.

3.2 Giv en effektiv algoritme der finder den rejse mellem to givne destinationer s og t der har færrest antal omstigninger. Angiv køretiden af din algoritme i asymptotisk notation. Din analyse skal være så tæt som mulig.

3.3 Firmaet ønsker også hjælp til at finde ud af hvornår busserne skal stoppe og tanke. Givet en busrute, kender vi placeringen af alle tankstationerne på ruten. Vi ved også hvor mange km k bussen kan køre på en fuld tank. Vi ønsker at lave så få stop for at tanke som muligt. Vi antager at bussen starter med en fuld tank. Vi kalder dette for *kør-og-tank* problemet.

Eksempel. Ruten er 300km. Bussen kan køre 110 km på en fuld tank. Tankstationerne på ruten ligger efter 10, 50, 80, 140, 180, 250, og 270 km.
I dette tilfælde skal bussen skal tanke mindst 3 gange, f.eks. efter 50, 140 og 250 km.

En af chaufførerne kommer med følgende strategi:

Grådig strategi: Kør så langt som muligt før der tankes.

a) Angiv hvor bussen stopper for at tanke hvis strategien bruges på eksemplet ovenfor.

b) Giv et argument for at den grådige strategi er korrekt eller giv et modeksempel.

Opgave 4 (rekursion)

4.1 Denne opgave omhandler en algoritme for fakultetsfunktionen ($n! = n \cdot (n-1) \cdots 1$). Nedenfor er 3 forsøg på at lave en rekursiv algoritme der beregner $n!$.

```
Algoritme Fak1(n)
  if n = 1
     return n
  x = n · (Fak1(n) − 1)
  return x
```

```
Algoritme Fak2(n)
  if n = 0
     return n
  x = n · Fak2(n − 1)
  return x
```

```
Algoritme Fak3(n)
  if n = 1
     return n
  return n · Fak3(n − 1)
```

Hvilken af algoritmerne beregner $n!$ korrekt når n er et positivt heltal: _____

4.2 Nedenstående algoritme tager et positivt heltal som input.

```
Algoritme Rekur(n)
1.   if  n ≤ 0
2.      return 0
3.   return n + Rekur(n − 1) + 2
```

Giv iterativ variant af algoritmen:

Opgave 5 (datastrukturer)

5.1 I hvilke af nedenstående datastrukturer kan man lave søgning i worst case $O(\log n)$ tid, hvor n er antal elementer i datastrukturen (du må gerne sætte mere end et kryds)?

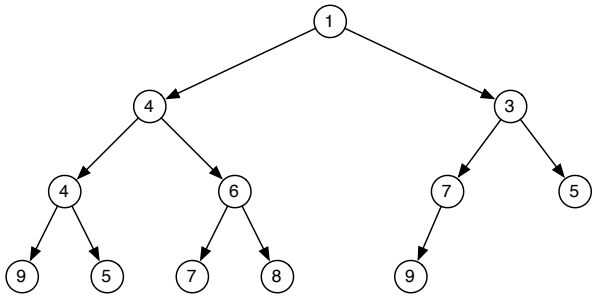
1 Hashtabel	2 sorteret hægtet liste	3 hob
4 sorteret tabel	5 usorteret tabel	6 binært søgetræ

5.2 I hvilke af nedenstående datastrukturer kan man lave indsættelse i worst case $O(1)$ tid, hvor n er antal elementer i datastrukturen (du må gerne sætte mere end et kryds)?

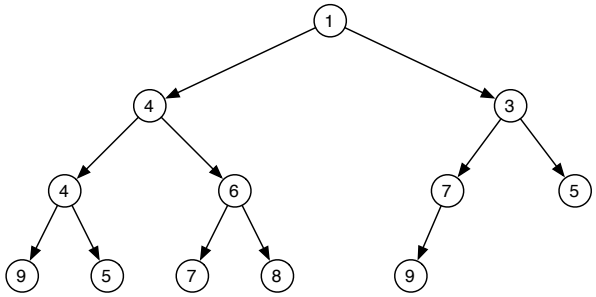
1 Hashtabel	2 usorteret hægtet liste	3 hob
4 sorteret tabel	5 binært søgetræ	

5.3 Denne opgave omhandler hobe, som beskrevet i bogen i afsnit 2.5.

a) Angiv hvordan hoben nedenfor ser ud efter indsættelse af et element med nøgle 2.



b) Angiv hvordan hoben nedenfor ser ud efter sletning af elementet med nøglen 6.



c) Giv en algoritme der i lineær tid checker om nøglerne i knuderne i et komplet binært træ opfylder hoborden (se side 59 i bogen for definitionen af hoborden).

5.4 I kør-og-tank problemet fra opgave 3.3 kender vi placeringen af tankstationerne på ruten. Antag at placeringen af en tankstation t er angivet som afstanden d_t fra starten til t. Givet en bus' position p og hvor mange km k den kan køre på en fuld tank ønsker vi at finde den tankstation der ligger længst væk af de tankstationer bussen kan nå inden den løber tør for diesel. Dvs. givet p og k ønsker vi at finde t således $d_t \leq p + k$ og d_t er størst mulig. Lad n være antallet af tankstationer. Hvilken datastruktur kan bruges til at finde det ønskede t i $O(\log n)$ tid givet p og k?