Skriftlig Eksamen
Algoritmer og Datastrukturer (DM507)

Institut for Matematik og Datalogi
Syddansk Universitet, Odense

Torsdag den 26. juni 2008, kl. 9–13

Alle sædvanlige hjælpemidler (lærebøger, notater, osv.) samt brug af lommeregner er tilladt.

Eksamenssættet består af 6 opgaver på 8 nummererede sider (1–8).
Fuld besvarelse er besvarelse af alle 6 opgaver.
De enkelte opgavers vægt ved bedømmelsen er angivet i procent.

Der må gerne refereres til algoritmer og resultater fra lærebogen inklusive øvelsesopgaverne. Henvisninger til andre bøger accepteres ikke som besvarelse af et spørgsmål.

Bemærk, at hvis der er et spørgsmål, man ikke kan besvare, må man gerne besvare de efterfølgende spørgsmål og blot antage, at man har en løsning til de foregående spørgsmål.

Husk at begrunde dine svar!

Opgave 1 (15%)

Spørgsmål a (5%): Udfør radix sort med radix 10 på tallene

747, 765, 544, 754, 431, 231, 222

Vis resultatet efter hver iteration. □

Spørgsmål b (10%): Angiv best-case og worst-case køretid samt køretiden for sorteret input for Insertion Sort, Merge Sort og Quicksort.

	Best-case	Worst-case	Sorteret input
Insertion Sort			
Merge Sort			
Quicksort			

Husk at begrunde dine svar. □

Opgave 2 (10%)

Spørgsmål a (10%): Angiv den asymptotiske rækkefølge af følgende funktioner.

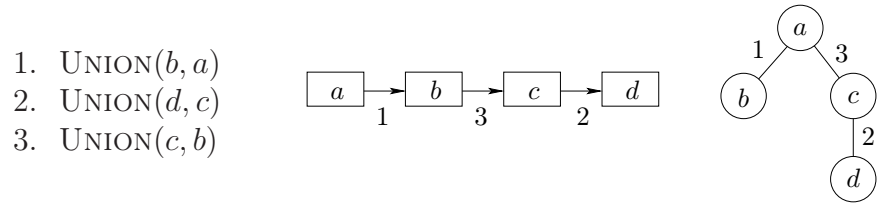
$$\sqrt{n},\ 2^n,\ (\log_{10} n)^2,\ n,\ \log_2 n$$

□

Opgave 3 (25%)

Denne opgave handler om datatypen disjunkte mængder.

> **Eksempel:** Betragt sekvensen af UNION-instruktioner i figur 1(a). I figur 1(b) vises den resulterende datastruktur, hvis man bruger lister til at repræsentere de disjunkte mængder. Tallene ud for de enkelte links angiver, hvilke UNION-instruktioner der forårsagede dem. Tilsvarende viser figur 1(c) resultatet, hvis man bruger træer til at repræsentere de disjunkte mængder.



Figur 1: Et lille eksempel

Spørgsmål a (6%): I dette spørgsmål antager vi, at de disjunkte mængder repræsenteres af hægtede lister, og at der bruges vægtet union. Hvis UNION(x, y) resulterer i, at to lister med samme vægt hægtes sammen, hægtes listen, som indeholder x, **efter** listen, som indeholder y (ligesom i eksemplet ovenfor).

Tegn den liste, som er resultatet af UNION-instruktionerne i figur 2. Angiv for hvert link, hvilken UNION-instruktion der forårsagede det, ligesom i eksemplet. □

Spørgsmål b (6%): I dette spørgsmål antager vi, at de disjunkte mængder repræsenteres af hægtede træer, og at der bruges union by rank (som defineret i lærebogen s. 508) men **ikke** path compression. Husk, at hvis UNION(x, y) resulterer i, at to træer med samme rang hægtes sammen, bliver roden i træet, som indeholder x, barn af roden i træet, som indeholder y (ligesom i eksemplet ovenfor).

1. UNION(b, a)
2. UNION(b, c)
3. UNION(e, d)
4. UNION(e, c)
5. UNION(g, f)
6. UNION(e, g)

Figur 2: UNION-instruktionerne brugt i spørgsmål a, b og c

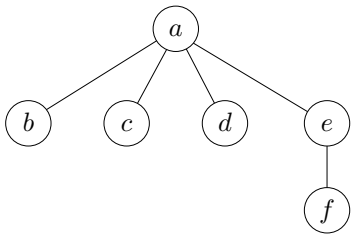
Tegn det træ, som er resultatet af UNION-instruktionerne i figur 2. Angiv for hvert link, hvilken UNION-instruktion der forårsagede det, ligesom i eksemplet. □

Spørgsmål c (6%): I dette spørgsmål antager vi, at de disjunkte mængder repræsenteres af træer, og at der bruges union by rank **og** path compression.

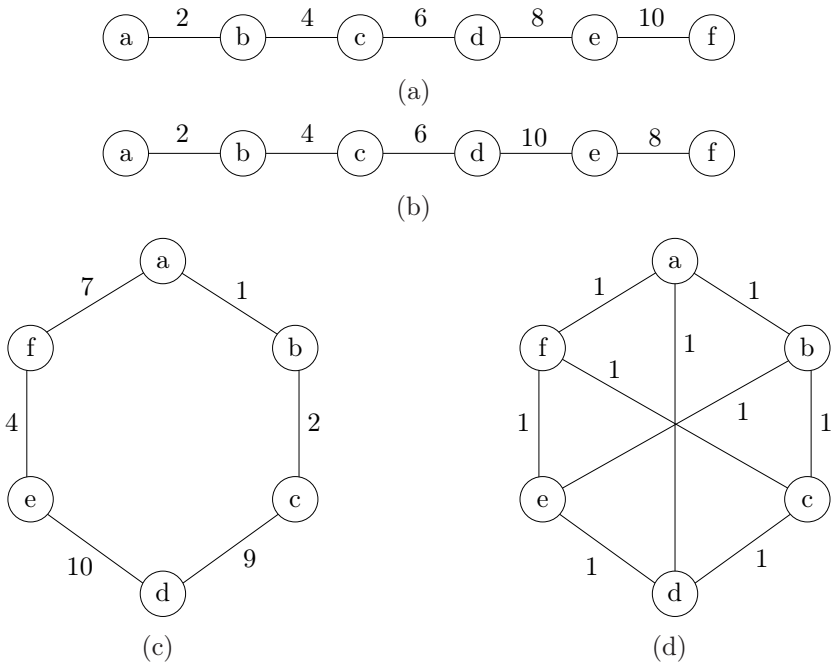
Tegn det træ, som er resultatet af UNION-instruktionerne i figur 2. Angiv for hvert link, hvilken UNION-instruktion der forårsagede det, ligesom i eksemplet. □

Spørgsmål d (7%): Betragt træet i figur 3, og antag at det er skabt vha. union by rank.

I figur 4 er vist fire vægtede grafer. Antag, at vi udfører Kruskals algoritme på hver af dem.
Hvilke af dem kunne resultere i træet i figur 3?
Husk at begrunde dit svar. □



Figur 3: Træet brugt i spørgsmål d



Figur 4: Fire vægtede grafer

Opgave 4 (10%)

Spørgsmål a (4%): Tegn alle mulige binære min-hobe, som indeholder fire knuder med prioriteterne 1, 2, 3 og 4. □

Spørgsmål b (6%): Tegn alle mulige binære søgetræer, som har højde 2 og indeholder fire knuder med nøglerne 1, 2, 3 og 4. □

Opgave 5 (15%)

I denne opgave skal vi se på en algoritme til at beregne a^b, hvor a og b er heltal. Vi skal bruge den binære repræsentation af b. Lad

$$b = 2^k b_k + 2^{k-1} b_{k-1} + \ldots + b_0$$

dvs. b_k er mest betydende bit i b, og b_0 er mindst betydende bit. Da kan vi repræsentere b vha. et array B af længde $k+1$, hvor $B[i] = b_i$.

Betragt følgende algoritme til at beregne a^b:

```
EKSP(a, B)
k ← length(B) − 1
c ← 1
for i ← k downto 0
    c ← c · c
    if B[i] = 1
        c ← c · a
return c
```

Spørgsmål a (10%): Lad B_{i+1} være tallet repræsenteret ved de $k-i$ mest betydende bits i b; dvs.

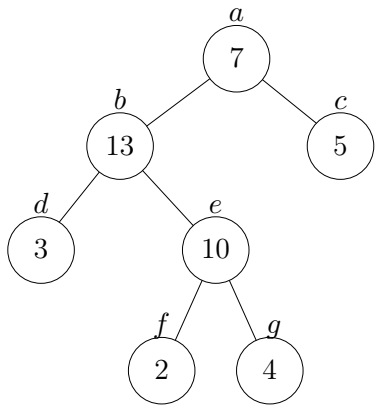
$$\begin{aligned}
B_{k+1} &= 0 \\
B_k &= b_k \\
B_{k-1} &= 2b_k + b_{k-1} \\
B_{k-2} &= 4b_k + 2b_{k-1} + b_{k-2} \\
&\vdots \\
B_0 &= 2^k b_k + 2^{k-1} b_{k-1} + \ldots + b_0 = b
\end{aligned}$$

Bevis følgende invariant for for-løkken:

I starten af for-løkken (dvs. lige efter opdateringen af i) er $c = a^{B_{i+1}}$

□

Spørgsmål b (5%): Brug resultatet fra spørgsmål a til at argumentere for, at algoritmen EKSP er korrekt. □



(a) Et binært træ med vægtede knuder

x	a	b	c	d	e	f	g
$W(x)$	18	13	5	3	10	2	4

(b) Vægten af en optimal sti-dækning for hvert undertræ

Figur 5: Et lille eksempel

Opgave 6 (25%)

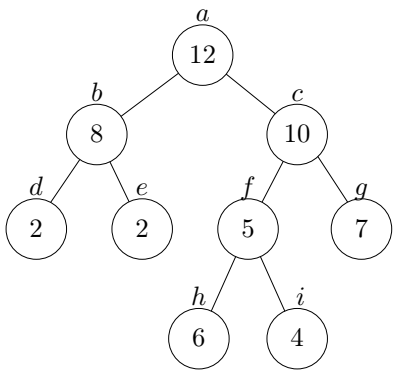
I denne opgave er der givet et binært træ, hvis knuder har vægte. Vægtene er positive heltal. Det gælder om at vælge en delmængde af knuderne med størst mulig samlet vægt. Hver sti fra roden til et blad skal indeholde præcis én af de valgte knuder.

En *sti-dækning* af et binært træ er en delmængde X af knuderne, som opfylder, at hver sti fra roden til et blad indeholder præcis én knude fra X. En optimal løsning er altså en sti-dækning med størst mulig samlet vægt.

Bemærk, at roden i et træ f.eks. udgør en sti-dækning af træet. Ligeledes udgør mængden af alle blade i træet en sti-dækning. Disse sti-dækninger er dog ikke nødvendigvis optimale sti-dækninger, dvs. de har ikke nødvendigvis størst mulig vægt.

> **Eksempel:** I figur 5(a) er vist et træ med vægtede knuder. Knuderne b og c udgør tilsammen en optimal sti-dækning. Det samme gør knuderne d, e og c. De to sti-dækninger har hver især samlet vægt 18.

For en knude x betegner $W(x)$ den samlede vægt af en optimal sti-dækning af undertræet med rod i x. I figur 5(b) er vist en tabel, som for hver knude x i træet angiver $W(x)$.



Figur 6: Et binært træ med vægtede knuder

Spørgsmål a (10%): Udfyld nedenstående tabel for træet vist i figur 6.

x	a	b	c	d	e	f	g	h	i
$W(x)$									

□

Spørgsmål b (5%): Betragt igen træet i figur 6. Angiv en sti-dækning med størst mulig samlet vægt. Dvs. angiv en knudemængde svarende til den vægt, du skrev i det første felt i tabellen. □

Spørgsmål c (10%): Angiv en algoritme baseret på dynamisk programmering, som givet et vægtet binært træ finder den største mulige vægt af en sti-dækning.

Hvad er din algoritmes køretid og pladsforbrug? □