UNIVERSITETET I TRONDHEIM
NORGES TEKNISKE HØGSKOLE
INSTITUTT FOR DATATEKNIKK OG TELEMATIKK
7034 Trondheim

Kontinuasjonseksamen i 45011 Algoritmer og Datastrukturer

Torsdag 17. august 1995, Kl. 0900-1300.

Hjelpemidler: Godkjent lommekalkulator samt alle trykte og håndskrevne hjelpemidler.
Faglig kontakt: Bjørn Olstad, tlf.: 73593447

Oppgave 1 (10 %)

Følgende PASCAL-kode er gitt:

```
VAR
    x : ARRAY[1..N] OF REAL;

PROCEDURE analyserA( i,j : INTEGER ) ; { Koden er ukjent }
PROCEDURE analyserB( i,j : INTEGER ) ; { Koden er ukjent }

PROCEDURE OppgaveA( indeks : INTEGER );
BEGIN
    IF indeks>1 THEN
    BEGIN
        analyserA(1,indeks);
        OppgaveA( indeks DIV 2 );
    END;
END;

PROCEDURE OppgaveB( start, slutt : INTEGER );
VAR
    midt : INTEGER;
BEGIN
    analyserB( start, slutt );
    IF slutt>start+1 THEN
    BEGIN
        midt := (start+slutt) DIV 2;
        OppgaveB( start, midt );
        OppgaveB( midt+1, slutt );
    END;
END;
```

"analyserA(i,j)" og "analyserB(i,j)" er her prosedyrer som kan gjøre ett eller annet med verdiene x[i],x[i+1],…,x[j].

1a) I et større programsystem har man satt som krav at kallet "OppgaveA(N);" skal ha en kjøretid som er maksimalt O(log(N)). Bestem ut fra dette den største kjøretiden som da kan tillates for prosedyren "analyserA(i,j)". Kjøretiden til "analyserA" skal uttrykkes som en funksjon av m der m=j-i+1. Begrunn svaret.

1b) I det samme programsystem har man satt som krav at kallet "OppgaveB(1,N);" skal ha en kjøretid som er maksimalt O(N*log(N)). Bestem ut fra dette den største kjøretiden som da kan tillates for prosedyren "analyserB(i,j)". Kjøretiden til "analyserB" skal uttrykkes som en funksjon av m der m=j-i+1. Begrunn svaret.

Oppgave 2 (10 %)

Det vises i læreboka at QuickSort er O(n^2). Forklar hvordan det er mulig å få QuickSort til å bli O(n*log(n)) ved å modifisere PARTITION rutinen. Forklar deretter hvorfor det ikke er vanlig å benytte den PARTITION rutina som gir best O-notasjon for QuickSort.

Oppgave 3 (10 %)

Forklar hvordan man i praksis kan gå fram for å velge mellom tellesortering og radixsortering når man vet hvor mange tall som skal sorteres og verdiområdet til disse tallene.

Oppgave 4 (10 %)

Det er gitt heltallene : 1, 4, 5, 7, 9, 10, 11, 13, 14, 16, 25, 27, 30.

4a) Tegn et binært søketre med minst mulig høyde som inneholder de gitte heltallene.

4b) Tegn et lovlig 23-tre som inneholder de gitte heltallene.

4c) Sett tallene 17, 19 og 20 inn i det 23-treet som ble laget i oppgave 4b. Tegn det nye 23-treet.

Oppgave 5 (20 %)

Beskriv en best mulig algoritme som kan finne den maksimale flyten fra alle løvnodene og opp til rotnoden i et binærtre. Anta at alle barnepekerene i binærtreet er merket med en verdi som angir den maksimale flyten fra barnet og opp til faren (kapasiteten til kanten mellom far og barn).

Hva blir kompleksiteten til den foreslåtte algoritmen som funksjon av antall noder i binærtreet?

Oppgave 6 (20 %)

Et program skal behandle måleverdier som foreligger på følgende form:

```
x  :  ARRAY[1..N] OF INTEGER;
```

Det kan antas at alle måleverdiene x[i] er heltall mellom 0 og 100. Det er kjent at måleverdiene inneholder en del støy. Samtidig vet man at uten denne støyen ville måleverdiene vært en monotont stigende tallsekvens. Man er derfor interessert i å finne en monotont stigende tallfølge som er nærmest mulig måleverdiene. Foreslå en raskest mulig algoritme som ut fra x[1..N] kan finne y[1..N] slik at:

(1) $y[1] \le y[2] \le y[3] \le \ldots \le y[N]$ og

(2) $(y[1]-x[1])^2 + (y[2]-x[2])^2 + \ldots + (y[N]-x[N])^2$ er minst mulig.

HINT: Du kan anta at også y[1..N] er begrenset til heltall mellom 0 og 100. Problemet kan da blant annet løses ved hjelp av dynamisk programmering.

Oppgave 7 (10 %)

Et bilde kan sees på som et 2-dimensjonal mønster:

bilde : ARRAY[1..N,1..N] OF INTEGER;

Diskuter om noen av tekstsøkealgoritmene i pensum kan generaliseres til å søke etter gitte mønstre i bilder. Et mønster vil selv være et lite bilde:

mønster : ARRAY[1..M,1..M] OF INTEGER;

Oppgave 8 (10 %)

Det er gitt følgende tallrekke :

0, 1, 1, 2, 1, 2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4, 1, ...

Beskriv en rekurrensformel som kan benyttes til å generere den gitte tallrekken.