

KONTINUASJONSEKSAMEN I FAG 45011

ALGORITMER OG DATASTRUKTURER

Onsdag 17. august 1994 kl.0900-1300

Faglig kontakt under eksamen: Bjørn Olstad/Øystein Grøvlen, tlf. 3447/3470

Alle trykte og håndskrevne hjelpemidler tillatt.

Godkjent lommekalkulator tillatt.

Merk: Alle svar skal skrives på tilviste plasser i oppgaveteksten.

Oppgave 1 (15%)

For prosedyrene nedenfor, gi ved hjelp av O -notasjon, verste tilfelle kjøretider som en funksjon av n .

a)

```
procedure matmpy ( $n$  : integer);  
  var  
     $i, j, k$  : integer  
  begin  
    for  $i := 1$  to  $n$  do  
      for  $j := 1$  to  $n$  do begin  
         $C[i, j] := 0$ ;  
        for  $k := 1$  to  $n$  do  
           $C[i, j] := C[i, j] + A[i, k] * B[k, j]$   
        end  
      end  
    end
```

b)

```
procedure mystery ( $n$  : integer);  
  var  
     $i, j, k$  : integer  
  begin  
    for  $i := 1$  to  $n - 1$  do  
      for  $j := i + 1$  to  $n$  do  
        for  $k := 1$  to  $j$  do  
          {some statement requiring  $O(1)$  time}  
        end  
      end  
    end
```

c)

```
procedure veryodd (n : integer);  
  var  
    i, j, x, y : integer  
  begin  
    for i := 1 to n do  
      if odd(i) then begin  
        for j := i to n do  
          x := x + 1;  
        for j := 1 to i do  
          y := y + 1;  
        end  
      end  
    end
```

d)

```
function recursive (n : integer) : integer;  
  begin  
    if n ≤ 1 then  
      return (1)  
    else  
      return (recursive(n - 1) + recursive(n - 1))  
    end
```

Oppgave 2 (40%)

a)

Gitt en generell trestruktur basert på følgende datastrukturer:

```
TYPE Node = RECORD  
  Verdi : INTEGER;  
  Mor : ^Node;  
  Barn : ^Node;  
  Sosken : ^Node;  
END;  
VAR Tre: ^Node;
```

Skriv prosedyren `SkrivUt(Rot: ^Node)` som skriver ut verdiene til alle nodene i treet.

b)

Gitt tallene: 92, 15, 23, 78, 38, 35, 69, 76, 2, 36. Disse tallene skal settes inn i en hash-tabell med 9 elementer. Kollisjoner blir løst med lenking.

Foreslå en egen hash-funksjon, og tegn tabellen slik den ser ut når alle tallene er satt inn.

Svar:

Hash-funksjon: $h(K)=$

Hash-tabell:

0
1
2
3
4
5
6
7
8

c)

Et binært søketre inneholder tall mellom 1 og 1000. Vi ønsker å søke etter tallet 140. Er tallfølgene nedenfor mulige besøks-sekvenser for nodene i treet? (Kryss av)

svar:

		Mulig	Umulig
a	710, 415, 90, 107, 232, 140		
b	783, 710, 694, 506, 415, 250, 140		
c	90, 783, 107, 661, 232, 120, 250, 140		
d	2, 874, 170, 64, 163, 140		
e	874, 2, 798, 743, 170, 729, 447, 203, 140		

d)

Vi skal utføre matrisemultiplikasjonen $A_1 \cdot A_2 \cdot \dots \cdot A_6$, der matrisene har dimensjonene (3×6) , (6×2) , (2×11) , (11×2) , (2×9) , (9×5) .

Algoritmen MATRIX-CHAIN-ORDER blir brukt til å finne den optimale parentessettingen. Fyll tallene som mangler inn i tabellene m og s.

Svar:

Fyll ut parentessettingen: A_1 A_2 A_3 A_4 A_5 A_6

e)

En datamengde inneholder de følgende tegnene med tilhørende frekvenser:

a: 12%	d: 25%
b: 14%	e: 27%
a: 22%	

Huffmann-koding skal brukes til å komprimere datamengden. Gi Huffmann-koder for tegnene.

Svar:

a:	d:
b:	e:
c:	

Hvor mye plass sparer man ved å bruke Huffmann-koder i stedet for koder med fast lengde?

Svar: _____ %

Oppgave 3 (20%)

Gitt en tabell $A[0..N-1]$ som er slik at det fins et tall t slik at tallfølgen $A[t], A[t+1], \dots, A[t+N-1]$ først er strengt voksende, så strengt minkende. (Alle indeksene er modulo N).

Skriv et effektivt program som finner største verdi i tabellen. Gi verste tilfelle kjøretid for programmet. Det er tilstrekkelig å benytte pseudo-kode.

Svar:

```
PROGRAM Stoerste;  
VAR A : ARRAY [0..N-1] OF INTEGER;
```

```
BEGIN
```

```
  {Initialiserer A}
```

```
  .
```

```
  .
```

```
  .
```

```
END.
```

```
Kjøretid:  $\Theta$  ( )
```

Oppgave 4 (25%)

Et veikart er representert ved en graf, $G=(V,E)$, der vekten på kanten (a,b) er distansen fra a til b , $F=(v_1, \dots, v_f)$ er de nodene som har bensinstasjoner.

Vi ønsker å finne $ROUTE(G, s, m, d, t)$, den korteste veien fra startposisjonen, s , til målet, m , når d er avstanden en bil kan kjøre på full tank, og t er avstanden bilen kan kjøre før fylling med det som nå er på tanken.

Foreslå en effektiv algoritme, og gi verste tilfelle kjøretid for algoritmen.

Benytt pseudo-kode og referer om mulig til kjente algoritmer.