

**Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap**



EKSAMENSOPPGÅVE I FAG MNFIT112 – ALGORITMER OG DATASTRUKTURAR

Fagleg kontakt under eksamen: Kjetil Nørvåg

Tlf.: 93440

Eksamensdato: 15. mai 2002

Eksamenstid: 09.00-15.00

Vekttal: 4

Tillette hjelpemiddel: Ingen

Språkform: Nynorsk

Tal på sider, bokmål: 11

Tal på sider, nynorsk: 11

Tal på sider, engelsk: 11

Sensurdato: 05. juni 2002

Merk! Studentane må primært gjere seg kjend med sensur ved å oppsøke sensuroppslaga. Evt. telefonar om sensur må rettast til instituttet eller sensurtelefonane. Eksamenskontoret vil ikkje kunne svare på slike telefonar.

Oppg ve 1 – Fleirvalssp rsm l (“multiple choice”) – 15 %

Bruk svararket sist i oppg vesettet til denne oppg va. Dersom du finn fleire alternativ som du synest passar, *set kryss for det eine som passer best*. For   unng  at gode tipparar vert l na, vil eit galt svar gje f rre poeng enn om oppg va ikkje vert svart p .

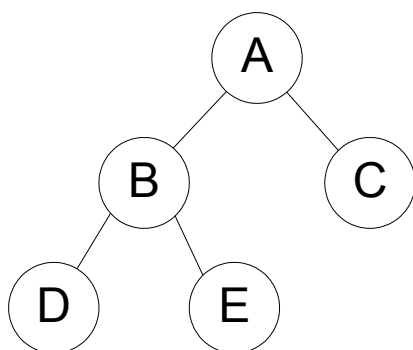
- a) Kva for ein av f lgjande p standar er *ikkje* korrekt?
- 1) Abstrakt datatype (ADT) er eit anna namn for datastruktur.
 - 2) Ein invariant er eit vilk r (“condition”) som alltid er sann p  eit visst punkt i ei algoritme.
 - 3) Ein l kke-invariant (“loop invariant”) er eit vilk r (“condition”) som er sann b de f r og etter utf ring av ei l kke.
 - 4) Ein b r unng  globale variable.
 - 5) Ein b r generelt bruke subprogram (prosedyrar, funksjonar, etc.) s  mykje som mogleg.
- b) Kva for ein av f lgjande p standar er *ikkje* korrekt?
- 1) Ein tabellvariabel (“array”) i Pascal er ei samling av element som har same datatype.
 - 2) Ein post (“record”) er ei gruppe med relaterte einingar (“items”) .
 - 3) Ein post kan ikkje vere eit felt i ein post.
 - 4) Ein tabellvariabel (“array”) i standard Pascal har statisk storleik.
 - 5) Ei dobbeltlenka liste tek st rre plass enn ei enkeltlenka liste.
- c) Effektiviteten/kompleksiteten til boble-sortering er:
- 1) $O(N)$
 - 2) $O(\log(N))$
 - 3) $O(N \log(N))$
 - 4) $O(N^2)$
 - 5) $O(2^N)$
- d) Effektiviteten/kompleksiteten til kvikksortering (“quicksort”) er:
- 1) $O(N)$
 - 2) $O(\log(N))$
 - 3) $O(N \log(N))$
 - 4) $O(N^2)$
 - 5) $O(2^N)$
 - 6) $O(M*N)$
- e) Ei listetraverserings-algoritme er:
- 1) $O(1)$
 - 2) $O(N)$
 - 3) $O(\log(N))$
 - 4) $O(N \log(N))$
 - 5) $O(N^2)$
 - 6) $O(M*N)$

f) Gitt følgende funksjon:

```
function F1(n : integer):integer;  
var sum,i,j,m : integer;  
begin  
    sum:=0;  
    m:=n;  
    for i:=1 to n do  
        for j:=1 to m do  
            sum:=sum+i+j;  
    F1:=sum;  
end;
```

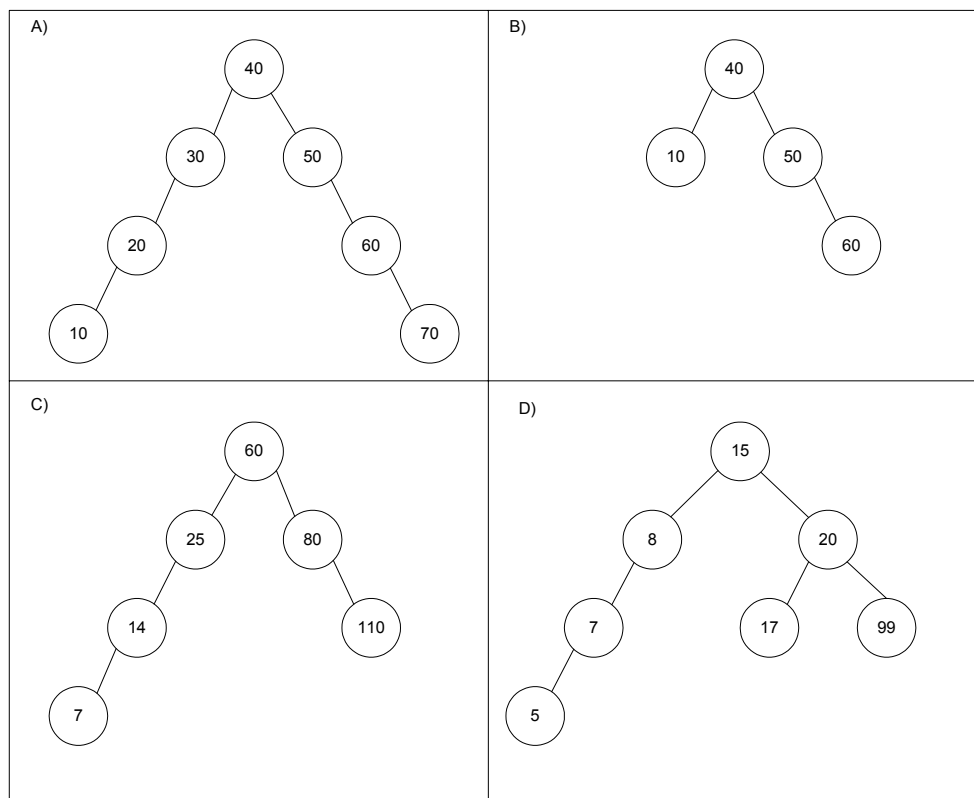
Kva er effektiviteten til denne funksjonen, uttrykt med O -notasjon?

- 1) $O(1)$
- 2) $O(N)$
- 3) $O(\log(N))$
- 4) $O(N \log(N))$
- 5) $O(N^2)$
- 6) $O(M*N)$



g) Kva for ein av dei følgjande påstandane om treet i figuren ovanfor er korrekt?

- 1) Treet er syklisk.
- 2) Treet er eit ikkje-fullt binært søketre.
- 3) Treet er eit fullt binært søketre.
- 4) Treet er balansert.
- 5) Treet har høgde lik 5.



h) Kva for eit av trea i figuren ovanfor er eit gyldig AVL-tre?

- 1) Tre A.
- 2) Tre B.
- 3) Tre C.
- 4) Tre D.

i) Gitt følgjande funksjon:

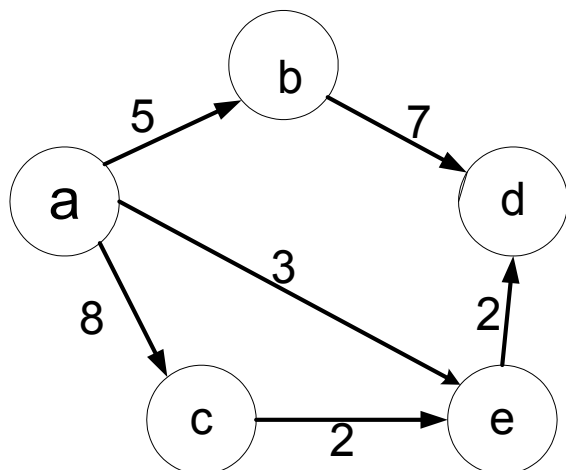
```

FunkX(v)
{v is a vertex}
  Mark v as visited
  for each unvisited vertex u adjacent to v do
  begin
    Mark the edge from u to v
    FunkX(u)
  end

```

Er dette mest sannsynleg pseudokode for:

- 1) Djupne-først søk?
- 2) Breidde-først søk?
- 3) Å finne spenntre?
- 4) Topologisk sortering?
- 5) Preorder traversering av eit tre?



- j) Kva for ein av dei følgjande påstandane om grafen G på figuren ovanfor er *ikkje* korrekt?
- 1) G er ein vekta graf.
 - 2) G er ein retta graf.
 - 3) G er ein syklisk graf.
 - 4) G er ein asyklisk graf.
 - 5) G er ikkje ein komplett graf.
- k) Kva for ein av desse operasjonane er *ikkje* ein typisk operasjon for ein kø-ADT?
- 1) $CreateQueue(Q)$
 - 2) $QueueIsEmpty(Q)$
 - 3) $QueueFront(Q)$
 - 4) $InsertQueue(Q, Position, Item)$
 - 5) $Remove(Q)$
- l) Kva for ein av desse operasjonane er *ikkje* ein typisk operasjon for ein stakk-ADT?
- 1) $CreateStack(S)$
 - 2) $StackIsEmpty(S)$
 - 3) $Push(S)$
 - 4) $TraverseStack(S)$
 - 5) $StackTop(S)$

Oppgave 2 – Rekursjon– 10 %

a) Gitt følgende program:

```

program Eksempel;
function EksFunk(n : integer):integer;
begin
  if n=1 then
    EksFunk := 0
  else
    EksFunk := EksFunk(n div 2) + 1
  end;
begin
  WriteLn('EksFunk(8)=' ,EksFunk(8))
end.

```

Kva vert utskrifta frå programmet?

b) Gitt funksjonen:

```

(1) function FX(x, n : integer) : integer;
(2) begin
(3)   if n=0 then
(4)     FX:=1
(5)   else
(6)     if n=1 then
(7)       FX:=x
(8)     else
(9)       if odd(n) then
(10)        FX:=FX(x*x, n div 2) *x
(11)      else
(12)        FX:=FX(x*x, n div 2)
(13) end;

```

Tips: odd(n) er ein Pascal-funksjon som returnerer *true* om n er eit oddetal, dvs. 1, 3, 5,...

Forklar kort kva FX kan brukast til. Er nokon av programlinjene overflødige?

Oppgave 3 – Søking og sortering – 15 %

- a) Kva er viktigaste føresetnaden for at binærsøk skal kunne brukast for å søke i ein tabell ("array")?
- b) Vis korleis binærsøk-algoritmen vert brukt for å søke etter talet 35 i tabellen nedanfor:

5	13	35	56	65	72	88	89	95	96	99
---	----	----	----	----	----	----	----	----	----	----

- c) Skisser flettesorteringsalgoritmen ("mergesort") med pseudokode. Vis korleis følgjande tabell ("array") med heiltal vert sortert med flettesortering:

54	23	68	22	92	99	78	12
----	----	----	----	----	----	----	----

- d) Kva er effektiviteten for flettesortering, uttrykt med O -notasjon? Grunnlegg svaret!

Oppgave 4 – Abstrakte datatypar– 30 %

- a) Forklar kort kva som er forskjellen mellom kø og stakk. Implementer deretter prosedyrane `CreateQueue` og `Add` i ein array-basert kø, med følgjande deklarasjonar som utgangspunkt:

```

const MaxQueue = 15;
type itemType = integer;
      queueType = record
          Items : array[1..MaxQueue] of itemType;
          Front, Rear : 1..MaxQueue;
          Count : 0..MaxQueue;
      end;
var Q : queueType;
procedure CreateQueue(var Q : queueType);
procedure Add(var Q : queueType; NewItem : itemType;
              var success : boolean);

```

- b) I Pascal er det mogleg å definere *sett*. For eksempel, om ein definerer ein variabel med type "set of 1..100" så kan dette sette settet innehalde heiltal mellom 1 og 100. Ein viktig eigenskap for eit sett, er at det ikkje inneheld duplikatverdiar. Til dømes er følgjande sant:

$$[2, 5, 17] + [8] = [2, 5, 8, 17]$$

$$[2, 5, 17] + [5] = [2, 5, 17]$$

Eit av problema med den innebygde støtta for sett i standard Pascal er at desse kun kan innehalde 256 forskjellige verdiar. De skal i denne oppgåva implementere ein meir generell sett-ADT. Desse setta skal kunne innehalde eit stort tal på element, og skal som eit minimum kunne støtte følgjande operasjonar:

```

procedure CreateSet(var S:setType; S1, S2:integer);
function MemberOfSet(var S:setType; SetItem:itemType):boolean;
procedure InsertIntoSet(var S:setType; SetItem:itemType);
procedure DeleteFromSet(var S:setType; SetItem:itemType);

```

I tillegg er følgjande deklarasjon gjeven:

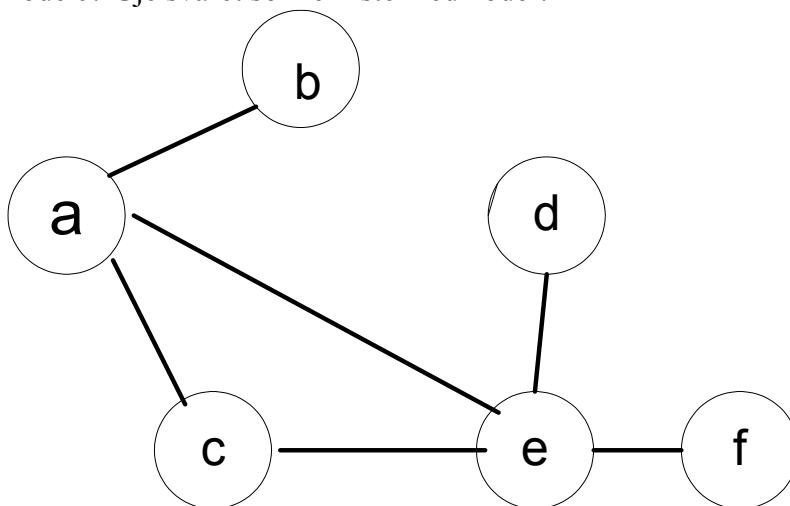
```
const NumItems = 600;
```

CreateSet vert brukt til å lage/initialisere eit tomt sett som kan ha element mellom S1 og S2, og der ein reknar med at det normalt vil vere ca. NumItems element (men det skal vere mogleg å legge inn nye medlemmar sjølv om det allereie er NumItems element i settet). Dvs. at CreateSet(S,1,10000) tyder det same som "set of 1..10000". Etter at kallet til CreateSet, vil settet S vere tomt. For å legge eit element til settet bruker ein InsertIntoSet, og for å fjerne eit element frå settet bruker ein DeleteFromSet. Ein reknar med at det normalt vil vere ca. NumItems=600 element i settet. MemberOfSet vert brukt for å teste om eit element er medlem i eit sett.

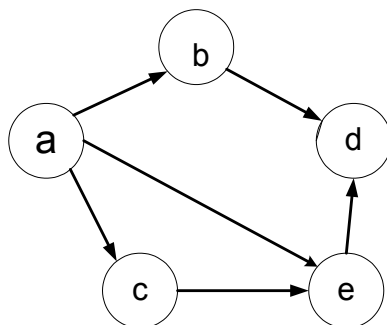
- 1) Beskriv kva datastruktur(ar) de vil bruke for å lagre eit sett i ein ADT som spesifisert ovanfor.
- 2) Deklarer nødvendige datatypar, datastrukturar og variablar, og implementer funksjonane og prosedyrane som er gitt. Gå utifrå at itemType er av type integer.
- 3) Kva er effektivitet (i O-notasjon) for dei forskjellige operasjonane?

Oppgåve 5 – Grafar– 15 %

- a) Kva vert resultatet av ei djupne-først ("depth-first") traversering av følgjande graf, med start i node c? Gje svaret som ei liste med noder.



- b) Kva vert resultatet av ei breidde-først ("breadth-first") traversering av grafen frå oppgåve a), med start i node c? Gje svaret som ei liste med noder.
- c) Kva er topologisk sortering av ein graf? Gje ei algoritme for topologisk sortering av ein graf. Kva vert resultatet om du bruker denne algoritmen på følgjande graf:

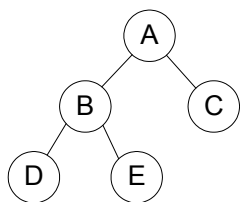


Oppg ve 6 –Tre og tabellar– 15 %

- a) Du skal bruke eit bin rt s ketre til   lagre heiltal. Start med eit tomt tre, og teikn for kvar av operasjonane under, i gjeven rekkef lgje, det resulterande bin rtreet:

- 1) Insert(4)
- 2) Insert(8)
- 3) Insert(6)
- 4) Insert(2)
- 5) Insert(9)
- 6) Insert(5)
- 7) Insert(3)
- 8) Remove(2)
- 9) Remove(4)

- a) Kva er gjennomsnittleg effektivitet for s k i eit bin rt s ketre med N noder? Grunngjev svaret!
- b) Kva er effektivitet i verste fall for s k i eit bin rt s ketre med N noder? N r skjer dette?
- c) Kva vert 1) resultatet av preorder traversering av treet i figuren nedanfor, og 2) resultatet av postorder traversering? (Operasjon p  ei node i treet ved traversering er utskriving av innhaldet av noda)



d) Du skal no bruke eit 2-3-tre for å lagre heiltal. Start med eit tomt tre, og teikn for kvar av operasjonane nedanfor, i gjeven rekkefølge, det resulterande 2-3-treet.

- 1) Insert(5)
- 2) Insert(40)
- 3) Insert(10)
- 4) Insert(20)
- 5) Insert(15)
- 6) Insert(30)
- 7) Remove(10)

e) Gå ut ifrå ein hash-tabell med 7 element/lenker, som skal brukast til å realisere ein tabell med heiltal. Kollisjonar vert løyst med lenking ("separate chaining"). Foreslå ein eigna hash-funksjon for denne tabellen, og set inn tala 24, 3, 10, 19, 25, 15. Teikn tabellen slik den ser ut etter at tala er sette inn.

Studentnr: _____ Studieprogram: _____ Arknr: _____ Antall ark: _____

Svarark for oppgåve 1 – Fleirvalsspørsmål ("multiple choice")

Dette arket skal brukast til å svare på oppgåve 1. Arket skal rives av oppgåvesettet og leverast.

Dersom du finn fleire alternativ som du synest passar set du kryss for det eine som passar best. For å unngå at gode tipparar vert løna, vil eit galt svar gje færre poeng enn om oppgåva ikkje vert svart på.

Alternativ→ Oppgåve↓	1	2	3	4	5	6
a)						
b)						
c)						
d)						
e)						
f)						
g)						
h)						
i)						
j)						
k)						
l)						