EKSAMEN I FAG 45011
ALGORITMER OG DATASTRUKTURER

Fredag 12. januar 1996, kl 0900 - 1300

Faglig kontakt under eksamen: Arne Halaas, tlf 93442

Alle kalkulatortyper tillatt. Alle trykte og håndskrevne hjelpemidler tillatt.

Merk: Avgrens oppgavebesvarelsen til totalt 4 sider.
Benytt en egen side (blank bakside) for hver av de 4 oppgavebesvarelsene.

Oppgave 1 (20 %).

Gitt den rekursive prosedyren:

```
Procedure tegning(x,y,L: real; n: integer);
if n > 0 then
begin
        linje(x+L/2, y);
        linje(x+L/4, y-L/4);
   tegning(x+L/4, y-L/4, L/2, n-1);
        linje(x, y-L/2);
        linje(x+L/2, y-L);  linje(x+L, y-L/2);  linje(x+L/2, y);
        linje(x+L, y);
        linje(x+L, y-L);  linje(x, y-L);  linje(x, y);
end;
```

Prosedyren **tegning** lager en sammenhengende, brukket strek på en tegnemaskin. Streken starter i punktet/koordinaten (x,y). Prosedyren linje(x0,y0) tegner et rett linjestykke fra løpende koordinat (der "pennen" befinner seg) fram til punktet (x0,y0).

(a) (5 %)

La S(L,n) være total streklengde (=summen av alle linjestykkers lengde) i tegningen.
Angi rekurrensligningen for å beregne S(L,n).

(b) (5 %)

Løs rekurrensligningen for S(L,n). Finn og kommenter verdien
$\lim_{n \to \infty} S(L,n)$.

(c) (10 %)

Vis hvilken tegning som produseres ved kallet: **tegning**(0,0,4,4).
Anta at enheten for L er centimeter.

Oppgave 2 (30 %)

Gitt heltalls-vektorene r(1..m) og s(1..n), der

r(1) + r(2) + ... + r(m) = s(1) + s(2) + ... + s(n).

(a) (15 %)

Vis hvordan maksimal-flyt-algoritmen (Ford-Fulkerson) kan benyttes til å avgjøre om en binærmatrise B(1..m,1..n) kan fylles med 1-ere og 0-ere slik at i-te rad får eksakt r(i) 1-ere , i=1,2,...,m, samtidig med at j-te søyle får eksakt s(j) 1-ere, for j=1,2,...,n.
Illustrer med en figur hvordan maksimal-flyt-algoritmen kan tilpasses i et tilfelle der m=3, n=4, r-vektoren består av verdiene (3,1,3) og s-vektoren består av verdiene (2,2,1,2).

(b) (15 %)

Beskriv, kort og punktvis, gjerne med en figur, hvordan du ville løse problemet i (a) mest mulig effektivt uten bruk av maksimal-flyt-algoritmen.

Oppgave 3 (30 %).

Se for deg en generell returmynt-automat, der en gitt myntenhet ikke nødvendigvis er et multiplum av en annen myntenhet som i vårt norske pengesystem.
Anta at et returbeløp B (et heltall) skal dekkes ved myntenheter M(1..k), der vi for enkelhets skyld lar minste myntenhet være M(1)=1.
For myntenhetene gjelder at M(1) < M(2) < ... < M(k).
Anta at det finnes ubegrensede mengder av hver myntenhet.

Målet vårt er å finne det minimale antall mynter som trengs for å dekke beløpet B.

(a) (10 %)

En grådighetsalgoritme for løsning av problemet er gitt ved:

```
While B > 0 do
begin
        Finn den  myntverdi M(j) som er slik at restverdien R : = B-M(j) blir minst  mulig;
        Sett B : = R
end;
```

Vis at denne grådighetsalgoritmen <u>ikke</u> garanterer at et minimalt antall mynter blir funnet.

(b) (20 %)

Et garantert minimalt antall mynter kan finnes ved dynamisk programmering. Her skal benyttes en tabell c(1..k,1..B), der c(i,j) er det minimale antall mynter som trengs for å dekke beløpet j ved kun å benytte mynter av typen M(1), M(2),..., M(i). Løsningen på hovedproblemet vil bli å finne i c(k,B).

Skriv Function Myntantall(B:Integer):Integer;.... som returnerer c(k,B) som sin verdi. Anta at vektoren M er globalt kjent.

Funksjonen bør ikke overskride 15 program-linjer. Merk her at vi for enkelhets skyld kun spør etter et totalt antall mynter, <u>ikke</u> hvilken mynt-fordeling som inngår i antallet. Programmeringsspråket kan gjerne være abstrakt og uformelt, men må være presist/selvforklarende.

<u>Oppgave 4 (20 %).</u>

Vi berører her to grunnleggende problemstillinger knyttet til teorien rundt NP-komplette problemer.

a) (10 %)

Anta at algoritmen A løser et NP-komplett problem Q.
Er det mulig å velge A til å løse et polynomisk problem R ?

Begrunn svaret ved enten å beskrive en situasjon der du kan gjøre et slikt valg, eller ved å forklare hvorfor et slikt valg ikke er mulig.

b) (10 %)

Gitt en urettet graf G=(V,E), der *u* og *v* er to forskjellige noder. Vi definerer problemet S:

S: Inneholder G en sti, med startnode *u* og endenode *v*, som går innom alle G's noder en og bare en gang,?

Ta stilling til påstanden: Det finnes <u>ingen</u> polynomisk-tid algoritme som løser problemet S med mindre P = NP. Begrunn.