Eksamen i fag
SIF8010 Algoritmer og Datastrukturer
Tirsdag 14. Desember 1999, kl 0900-1500

Faglig kontakt under eksamen: Arne Halaas, tlf. 73 593442.
Hjelpemidler: Alle kalkulatortyper tillatt. Alle trykte og håndskrevne hjelpemidler tillatt.
Rubrikksvar: Alle svar skal avgis i angitte svar-ruter. Ikke legg ved ekstra ark som svar.
Krav: Det kreves "bestått" både på de ordinære og på de øvingsrelaterte spørsmål.
Husk: Fyll inn rubrikken "Student nr." øverst på alle ark.

Oppgave 1. (18%)

a) Er høyden på et 2-3–tre med n dataposter $\boldsymbol{\theta}(\log_3 n)$?

b) Er Dijkstras algoritme basert på Dynamisk Programmering?

c) Er Kruskals algoritme en grådighetsalgoritme som alltid gir beste løsning?

d) Er antallet ordninger ved topologisk sortering av en vilkårlig graf $G(V,E)$ $\mathbf{O}(|V|!)$?

e) Er antallet ordninger ved topologisk sortering av en vilkårlig graf $G(V,E)$ $\mathbf{O}(|E|!)$?

f) Er Quicksort $\boldsymbol{\Omega}(n \cdot \log n)$?

g) Bør "sortering ved innsetting" brukes i forbindelse med Quicksort ?

h) Kan maks-flyt-algoritmen brukes til å finne ut om en sammenhengende graf G har en bro?

i) Er Huffmans algoritme aktuell i forbindelse med fletting (Merge) av $m > 2$ sorterte lister?

Svar: (Stryk "Ja" eller "Nei". Begrunnelsen må fylles ut. Hvert delsvar teller 2%)

a) Ja/nei	Begrunnelse
b) Ja/nei	Begrunnelse:
c) Ja/nei	Begrunnelse:
d) Ja/nei	Begrunnelse:
e) Ja/nei	Begrunnelse:

f) Ja/nei Begrunnelse:

g) Ja/nei Begrunnelse:

h) Ja/nei Begrunnelse:

i) Ja/nei Begrunnelse:

Oppgave 2. (20%)

Vi definerer problemet P(A,n,k,b) slik: *Finn, om mulig, et utvalg av k* (>1) *verdier i $A=\{a_1,a_2,...,a_n\}$ som er slik at summen av disse k verdiene er lik b. Alle verdiene er heltall.*

(a) Skisser en algoritme Q som løser problemet P(A,n,k,b) i $\mathbf{O}(n^{k-1} \log n)$ tid.

Svar: 4%

(b) Skisser en mer effektiv algoritme R som løser problemet P(A,n,k,b) i $\mathbf{O}(n^{k-1})$ tid når k>2.

Svar: 4%

(c) Hvordan vil du løse problemet P(A,n,k,b) når $n > 100$ og $k = n–3$?

Svar: 5%

(d) For hvilke verdier av k vil du anta at P(A,n,k,b) krever mest tid for å bli løst? (Begrunn.)

Svar: 4%

Vi definerer nå problemet P'(A,n,b) slik: *Finn, om mulig, et utvalg av inntil n verdier i* $A=\{a_1,a_2,...,a_n\}$ *som er slik at summen av disse verdiene er lik b. Alle verdiene er heltall.*

(e) Hvilken metode vil du foreslå for å løse problemet P'? Angi metodens tidskompleksitet.

Svar: 3%

Oppgave 3. (8%)

Student Lurvik hevder å ha utviklet en ny datastruktur for prioritetskøer som støtter operasjonene `Insert(Queue,element)`, `FindMaximum(Queue)` og `DeleteMaximum(Queue)`. Lurvik påstår at alle disse 3 operasjonene kun krever **O**(1) tid.

(a) Det er ingen grunn til å tro på Lurvik. Hvorfor ikke?

Svar: 8%

Oppgave 4. (21%)

Vi skal her se på et problem som skal løses ved hjelp av Dynamisk Programmering:

Problem P(S, n): *Gitt en sekvens* $S = \langle s_1,s_2,\ldots,s_n\rangle$ *bestående av n heltall. Finn lengden* L_n *av den lengste subsekvensen* $S^\bullet$ *i S som er slik at verdiene i* $S^\bullet$ *er stigende.Verdiene i* $S^\bullet$ *må ikke nødvendigvis være naboer i S.*
Merk at det her kun spørres etter lengden L_n av den (en av de) lengste subsekvensen(e) i S.

Eksempel (n=9):
$S = \langle 9,5,2,8,7,3,1,6,4\rangle$. Her er $L_n = 3$. Subsekvensen $S^\bullet$ består da av enten $\langle 2,3,4\rangle$ eller $\langle 2,3,6\rangle$.

(a) Beskriv kort hvordan vi kan finne L_n ved dynamisk programmering.

Svar: 5%

(b) Finn tidskompleksiteten til metoden foreslått i (a)

Svar: 4%

(c) Forklar kort hvordan du kan finne selve sekvensen $S^\bullet$ ved å føye ekstra informasjon til løsningen foreslått i **(a)**. Bruk gjerne det oppgitte eksempelet for å illustrere ideene.

Svar: 4%

(d) Hva blir tidkompleksiteten i **(c)**?

Svar: 4%

(e) Foreslå en praktisk sammenheng der problemet P(S,n) er av interesse.

Svar: 4%

Oppgave 5. (8%)

Vi skal i denne oppgaven se på et praktisk problem knyttet til et rettet nettverk *G*=(*V*,*E*). Kantene i *E* representerer vannførende kanaler, hver med en spesifisert kapasitet *c* kubikkmeter pr. sekund. Kanalene møtes i noder som ikke har noen kapasitetsbeskrankning. Kantene har i tillegg en parameter *InTown* som har verdien *True* dersom kanten ligger i tettbebyggelsen, *False* ellers.

(Vi antar at en eventuell oversvømmelse bare vil forekomme i én kanal, dvs. vi ser bare på hvor oversvømmelsen starter.)

(a) Hvilken metode vil du bruke for å finne ut om en det er mulig at en oversvømmelse rammer en av kanalene i tettbygd strøk?

Svar: 4%

(b) Hvordan vil du finne ut om en oversvømmelse garantert vil ramme et tettbygd strøk?

Svar: 4%

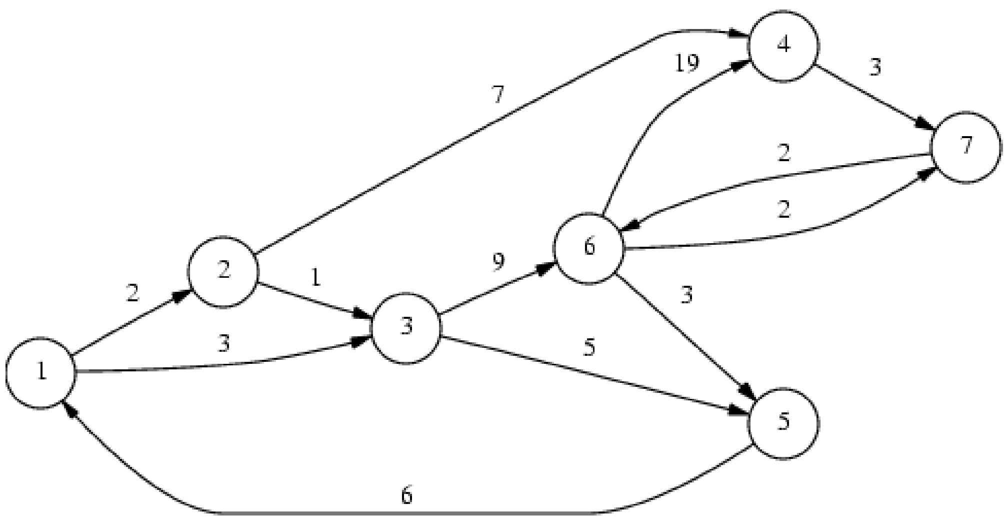
Oppgave 6, Øvingsrelaterte oppgaver. (25%)

(a) Kjør Partition på tallene : 13, 7, 11, 4, 6, 2, 0, 32, 29.
Bruk tallet 13 som pivot-element. Vis høyre og venstre partisjoneringsindeks per steg (i,j).

Svar: 4%

(b) Hva ville vært optimalt pivot-element generelt sett når Partition blir brukt i Quicksort?

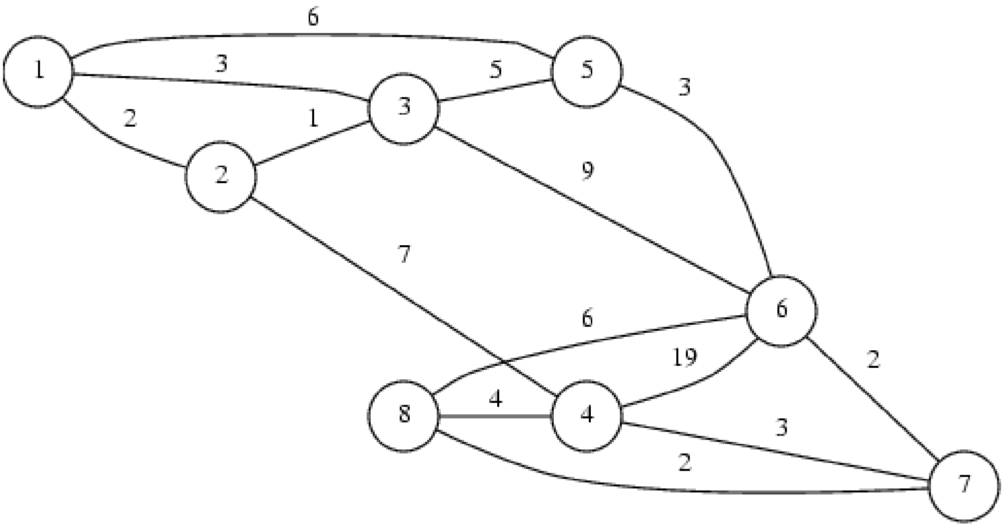
Svar: 2%



(c) Bruk Dijkstras algoritme til å finne korteste vei fra node 1 til de andre nodene i grafen over. Fyll inn verdier for avstandsfunksjonen d (for hvert steg i algoritmen) i tabellen under:

Svar: (8%)

Steg	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7
1							
2							
3							
4							
5							
6							
7							



(d) Finn minste spenntre i grafen over, ved bruk av Prims algoritme. Fyll inn nodepar (fra-node – til-node) for kantene du legger til i hvert steg i tabellen under:

Svar: 6%

Steg	Fra-node	Til-node
1		
2		
3		
4		
5		
6		
7		
8		

(e) Hva blir summen av kantene i det minimale spenntreet?

Svar: 2%

(f) Kan man ha et største spenntre? Hvordan vil du evt. finne det, og hva blir tidskompleksiteten?

Svar: 3%