Eksamen i fag
SIF8010 Algoritmer og Datastrukturer
Tirsdag 18. Desember 2000, 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. (20%)

Følgende 10 påstander skal besvares og begrunnes:

a) Alle algoritmer MinMax for å finne både et største og et minste element i en to-dimensjonal heltalls-tabell med **n** rader og **n** kolonner har tidskompleksitet $\Omega(n^2)$.

b) Sortering ved innsetting (n elementer) har tidskompleksitet $\Omega(n)$.

c) Boblesortering (**n** elementer) har tidskompleksitet $O(n^2 \log n)$.

d) Spredt lagring (hash-teknikk) kan effektivt brukes ved sortering.

e) Dijkstras algoritme kan brukes til å finne alle-til-alle (noder) korteste vei.

f) Prefiks-traversering av et binært søketre svarer til sortert rekkefølge.

g) I Ford-Fulkerson's algoritme kan en velge å gi kantene både en øvre og en nedre flytgrense > 0.

h) Dersom en graf **G=(V,E)** er et tre, kan en i **O(1)** tid avgjøre om **G** er tofargbar.

i) Kruskals og Prims algoritmer gir alltid like korte (minimale), men generelt forskjellige spenntrær.

j) Dersom et gitt problem i sin generelle form er **NP**-komplett, har det ingen hensikt å prøve og løse dette når problemets størrelse øker over en viss terskel.

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:

j) Ja/nei Begrunnelse:

Oppgave 2. (15%)

Et meget stort datasett (forskjellige positive heltall) **T[1..n]** skal gjøres tilgjengelig på internett for mange hyppige brukere. Datasettet oppdateres ukentlig. Du blir gitt oppgaven å lage en så effektiv som mulig algoritme som gjør følgende:

> Innverdier (kun heltall): **T,n,a** og **b** (der **a** og **b** er gitt av brukeren)
> Utverdier (heltall): **k** og **M** = Medianen (midtverdien) blant alle **k** verdier **{T[i]}** som tilfredsstiller kravet **a ≤ T[i] ≤ b**.

Svar (algoritmeskisse): 15%

Tidskompleksitet: O( )
Begrunnelse for algoritmevalget:

Oppgave 3. (20%)

Du har gitt **n** punkter i **x-y**-planet, **P1:(x1,y1),P2:(x2,y2),...,Pn:(xn,yn)**. Du skal finne en beste vei fra **P1** til **Pn**, via et utvalg av de øvrige **(n-2)** punktene, som er slik at den lengste avstanden mellom nabopunkter på veien er så kort som mulig. (Tenk deg at du skal hoppe på stener over en elv og at det lengste hoppet skal være så kort som mulig.)

(a) Skisser en så effektiv som mulig algoritme som løser dette beste-vei problemet ved å bruke Dijkstras algoritme (uendret) som subrutine.

Svar: 8%

Algoritmens tidskompleksitet: O( )
Begrunnelse:

(b) Finn en mer effektiv algoritme for å løse vårt beste-vei-problem ved å modifisere koden i Dijkstras algoritme. Vis modifikasjonene i detalj.

Svar: 12%

Algoritmens tidskompleksitet: O( )
Begrunnelse:

Oppgave 4. (20%)

La **G=(V,E)** være en sammenhengende, urettet og vektet graf. Anta at kantenes vekter er forskjellige positive heltall. Vi benevner **G**'s minimale spenntre for **MST**.

(a) Anta at vi øker alle vektene for kanter i **E** med en konstant verdi **c > 0**. Er **MST** fortsatt et minimalt spenntre for **G**? Hvis ikke, hvordan finner vi mest effektivt **G**'s nye spenntre?

Svar: 6%

(b) La nå **e** være en vilkårlig kant i **E**, og la **T(e)** være det spenntre i **G** som har minst kostnad av alle spenntrær som inneholder kanten **e.** Beskriv en så effektiv som mulig algoritme som finner **T(e)** for samtlige **|E|** kanter **e** ε **E.** Finn også den foreslåtte algoritmens kompleksitet.

Svar: 14%

Algoritmens kompleksitet: O ( )

Oppgave 5 (Øvingsrelatert) (25%)

a) Hvor effektivt kan vi sortere n tall i tallområdet $0..n^n$? Begrunn svaret.

3%:

b) Hva er tidskompleksiteten til 0-1 ryggsekk-problemet? Begrunn svaret.

2%:

c) Hvorfor har tidskompleksiteten til innsettings-sortering samme *worst-case* og *average-case*?

2%:

d) Når er det lurt å bruke memoisering?

2%:

e) Vil du bruke *dybde-først-søk* eller *bredde-først-søk* for å finne én-til-alle korteste vei i en graf der alle kantene har vekt 1? (Begrunn.)

2%:

f) Gi en praktisk anvendelse av *største uavhengige delmengde*.

3%:

g) Hvis du har et sett $\{s_i\}$ med n strenger av ulik lengde, der lengden av strengen s_i er $l(s_i)$, hva er kjøretiden for en mest mulig effektiv algoritme som sorterer strengene?

3%:

h) Hva er sammenhengen mellom redigerings-avstand (*edit distance*) og lengste felles subsekvens (*longest common subsequence*)?

4%:

i) I en bok leser vi at 0-1 ryggsekk-problemet er NP-komplett. Hvordan harmonerer dette med ditt svar på spørsmål b) ?

4%: