Kontinuasjonseksamen i fag
SIF8010 Algoritmer og Datastrukturer
Torsdag 9. August 2001, 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. Stryk over det svaret (JA eller NEI) du mener er galt.
Krav: Det kreves "bestått" både på de ordinære og på de øvingsrelaterte spørsmål, Oppg. 15-20.
Husk: Fyll inn rubrikken "Student nr" øverst på alle ark.

OPPGAVE 1. (3%)

Påstand: Når vi bruker uttrykket "algoritme A er O(f(n))" er det underforstått at vi sikter til gjennomsnittlig kjøretid ("average case") for algoritmen A.

Svar: NEI
Begrunnelse: Vi underforstår "worst case" fordi både "average case" og "best case" da blir omfattet.

OPPGAVE 2. (3%)

Påstand: $\theta(\ln n) = \theta(\lg n)$.

Svar: JA
Begrunnelse: Logaritmen til et tall med ett grunntall skiller seg fra logaritmen til et tall med et annet grunntall kun med en konstant: $\log_c n = (\log_b n) * (\log_c b)$. Her er c=e og b=2.

OPPGAVE 3. (3%)

Påstand: $n = O(n^2)$.

Svar: JA
Begrunnelse: Trivielt er f(n)=n med i den mengden av funksjoner som har n^2 som dominerende faktor.

OPPGAVE 4. (4%)

Påstand: Det er mer nyttig å vite at en algoritme er $\theta(g(n))$ enn at den er $O(g(n))$.

Svar: JA
Begrunnelse: Da vet en noe om "best case" i tillegg til "worst case".

OPPGAVE 5. (4%)

Påstand: Sortering ved innsetting utnytter kunnskap om verdiområdet til de verdier som skal sorteres.

Svar: NEI
Begrunnelse: Metoden benytter kun nøkkel-sammenligninger og virker derved like godt dersom en blander (svært) små og (veldig) store tall i sorteringen.

OPPGAVE 6. (4%)

Påstand: Alle trestrukturer med n løvnoder har høyde O(lg n).

Svar: NEI
Begrunnelse: Enkelte trær kan degenerere til å bli like effektiv som en liste. Eksempel:
Et binært søketre der en setter inn verdiene i (omvendt) sortert rekkefølge.

OPPGAVE 7. (4%)

Påstand: QUICKSORT er den beste sorteringsmetoden når en skal sortere desimaltall (flyttall).

Svar: NEI
Begrunnelse: Dette kan en ikke si generelt. Dersom desimaltallene f.eks. er slik at heltallsdelen effektivt kan brukes til å splitte datamengden i undermengder vil en lett utkonkurrere Quicksort, som sorterer "på stedet" ("in situ"), kun med bruk av nøkkelsammenligninger.

OPPGAVE 8. (6%)

Påstand: Når vi trenger binomialkoeffisientene n! / (k! (n-k)!), der n = 0,1,2,...,N,
kreves det $O(N^3)$ tid å beregne disse 1. gang, men O(1) tid å finne en koeffisient senere.

Svar: JA/NEI
Begrunnelse: Det tar $O(N^2)$ tid å beregne binomialkoeffisientene (Pascal's trekant.)
Når koeffisientene er beregnet tar det konstant tid å finne en koeffisient ved tabelloppslag.
Dermed er ikke $O(N^3)$ en tett grense, og det er urimelig å si at det "kreves $O(N^3)$ tid."
Likevel er det korrekt, men man bør da påpeke at $N^2 = O(N^3)$.

OPPGAVE 9. (6%)

Påstand: For å finne korteste vei i asykliske grafer, der negative kantlengder tillates,
er Bellman-Ford's algoritme best egnet.

Svar: NEI
Begrunnelse: Her skal en bruke DAG-Shortest-Path, den er i dette tilfellet enklere & mer effektiv enn B-F's algoritme.

OPPGAVE 10. (6%)

Påstand: Kjøretiden for en algoritme som avgjør om et tre G = (V,E) er tofargbart (d.v.s.: nabonoder skal gis forskjellig farge.) er $\Omega(|V|)$

Svar: NEI
Begrunnelse: Et tre er alltid tofargbart. Dersom en vet at G er et tre kan en derfor i O(1) tid avgjøre om treet er tofargbart. (En nodes barn- og foreldre-noder gis annen farge enn noden selv.)

OPPGAVE 11. (8%)

Påstand: Verdien x^n, der x er et flyttall og n et stort heltall, beregnes effektivt ved Dynamisk Programmering.

Svar: JA
Begrunnelse: En benytter rekursive egenskaper som $x^n = x^{n/2}\ x^{n/2}$
og finner & lagrer de forskjellige 2-potensene av x. Dersom en senere trenger f.eks. x^{13} kan en ved å slå opp i en tabell lett finne $x^{13} = x^8\ x^4\ x^1$ ved 2, istedet for 12, multiplikasjoner.

OPPGAVE 12. (8%)

Påstand: Dersom vi i grafen G = (V,E) vil undersøke om det finnes en node v som har inngående kanter fra samtlige øvrige noder, inklusive seg selv, kan dette gjøres med en $\Omega(|V|)$-algoritme. Noden v skal ialt ha |V| kanter.

Svar: JA
Begrunnelse: Dette problemet tilsvarer "kjendis-problemet", ref. Oppg. 2, dato 13.12.1997. ($\theta(|V|)$, dermed trivielt $\Omega(|V|)$.)
Merk: Alle løsbare problemer kan løses med en $\Omega(|V|)$-algoritme.

OPPGAVE 13. (8%)

Påstand: 0/1 Ryggsekkproblemet (Knapsack) er NP-komplett.

Svar: JA
Begrunnelse:
Det vises til kompendiet og til forelesninger om Dynamisk Programmering, der det vises at problemet kan løses i O(nW) tid. En dobling av W betyr en dobling av øvre grense.

OPPGAVE 14. (8%)

Påstand: Når vi har funnet en maksimal flyt i et nettverk finner vi samtidig også nettverkets "flaskehals" (et entydig såkalt minimalt snitt) der kapasitetsøkning eventuelt kan foretas.

Svar: JA & NEI
Begrunnelse: Dette kan være både riktig og galt. Det er riktig dersom det finnes kun ett snitt med minimal kapasitet. Det er galt dersom det finnes mer en ett minimalt snitt.

OPPGAVE 15. (5%)

Påstand: n tall i tallområdet 0 til n*log(n) kan sorteres i O(n) tid.

Svar: JA
Begrunnelse: *n* tall i tallområdet 0 til n^2 kan sorteres i O(*n*) tid med radiks-sortering (2 siffer). Tallområdet 0 til *n**log(*n*) faller innenfor dette.

For at dette skal være riktig, må det antas at tallene er heltall.

OPPGAVE 16. (5%)

Påstand: Memoisering er mer effektivt enn vanlig dynamisk programmering.

Svar: NEI
Begrunnelse: Memoisering virker på samme måte som vanlig (iterativ) dynamisk programmering, men gir ekstra arbeid i forbindelse med rekursive funksjonskall etc.

OPPGAVE 17. (5%)

Påstand: Et Huffman-tre der nodene har frekvenser
2, 4, 8, ..., 2^n vil ha høyde θ(n).

Svar: JA
Begrunnelse: Siden summen av alle elementer lavere enn et gitt element alltid vil være mindre enn det gitte elementet, vil treet være helt ubalansert.

OPPGAVE 18. (5%)

Påstand: Dijkstras algoritme kan ikke brukes på grafer som inneholder negative kanter fordi den vil havne i en uendelig løkke.

Svar: NEI (mao.: påstanden er gal)
Begrunnelse: Negative kanter (såfremt det også er positive kanter i grafen) vil gi galt svar. Man kan aldri være sikker på at noden med lavest estimat har korrekt estimat.

OPPGAVE 19. (5%)

Påstand: n heltall representert med totalt m bits kan sorteres i O(n*log(m)) tid.

Obs: Grunnet trykkfeil vil denne oppgaven ikke telle ved sensurering.
Riktig oppgavetekst skulle være:
"*n* heltall representert med totalt *m* bits kan sorteres i $O(m*\log(n))$ tid." Svaret ville ha vært "ja" siden tallene kan sorteres med radiks-sortering for tall med ulikt antall siffer i $O(m)$ tid.

Slik teksten står vil det kreve et motbevis med en vanskelighetsgrad som går ut over intensjonene med oppgaven.

OPPGAVE 20. (5%)

Påstand: Topologisk sortering av en komplett graf med n noder tar $\theta(n^2)$ tid.

Svar: JA/NEI
Begrunnelse: (Hvis JA)
Tolker "komplett graf" som en rettet, asyklisk graf der den underliggende urettede grafen er komplett. Topologisk sortering har kjøretid $\theta(|E| + |V|)$; kjøretiden blir dermed $|E| = \theta(n^2)$, der $n = |V|$.

Begrunnelse: (Hvis NEI)
1. En komplett graf er per definisjon urettet og kan dermed ikke sorteres topologisk.
2. En komplett, rettet graf vil ha kanter begge veier mellom alle noder. Siden dette innebærer at grafen har løkker, vil topologisk sortering være umulig.

Benytt plassen nedenfor til eventuelle kommentarer:

Merk: Vekt-tallene gir totalt 105%. Siden oppgave 19 er strøket vil den totale vektleggingen likevel ende på 100%.