Eksamen i fag
TDT4120 Algoritmer og Datastrukturer
Tirsdag 3. August 2004, kl 0900-1500

Faglig kontakt under eksamen: Magnus Lie Hetland tlf. 91851949
Hjelpemiddel: Alle kalkulatortyper tillatt. Alle trykte og håndskrevne hjelpemidler tillatt. Skriv svarene i de oppgitte rutene. Et svar uten begrunnelse teller ikke. En kan ta i bruk tilleggsark dersom dette er nødvendig. Før på "Student nr." på hvert svarark.

Oppgavene er merket #n/p, der n er oppgavenummeret og p er antall poeng som maksimalt kan oppnås når både svaret (Ja /Nei) og tilhørende begrunnelse er riktig.

Det kan maksimalt oppnås 50 poeng for besvarelsen.

#1/1: "Best case" kjøretid for INSERTION SORT ved sortering av n elementer er O(n).

Svar: Ja Begrunnelse: Gjelder når data er sortert på forhånd

#2/3: Ved å bruke Master-teoremet finner en løsningen T(n) = Θ (nlogn)
på rekurrensen T(n) = 3 T(n/3) + logn

Svar: N ei Begrunnelse: Master-teoremet gir T(n) = Θ (n)

#3/3: For et vilkårlig binært søketre med n noder kan vi skrive ut nodene i sortert rekkefølge på O(n) tid.

Svar: Ja Begrunnelse: For hver node vil INORDER-TREE-WALK rekursivt bli kalt eksakt 2 ganger – en gang for nodens høyre og en gang for nodens venstre subtre.

#4/1: Ethvert binært søketre med n noder har høyde O(logn).

Svar: Nei Begrunnelse: Settes nodene inn i f.eks. sortert rekkefølge blir høyden O(n).

#5/1: Enhver haug (heap) som benyttes av HEAPSORT for å sortere n elementer har høyde O(log n)

Svar: Ja Begrunnelse: Haugens delvis sorterte tre har alltid alle terminalnoder på høyst 2
forskjellige nabonivåer. Dette er det viktigste aspekt ved metodens fordeler.

#6/2: Haugen i HEAPSORT er tilfeldig ordnet.

Svar: Nei Begrunnelse: En node er alltid dominant (eller dominert av) begge sine undernoder, avhengig av om største/minste nodeverdi skal svare til rotens verdi. Den orden som gjelder kalles gjerne et "rettet tre".

#7/2: Det er slik at $n\log n^2 = O(n^2)$.

Svar: Ja Begrunnelse: $n\log n^2 = 2n\log n = O(n\log n)$ og da også i $O(n^2)$.

#8/3: MERGESORT bruker i worst-case $O(n^2)$ tid

Svar: Ja Begrunnelse: MERGESORT kjører i O(n logn) tid, altså da også i $O(n^2)$ tid.

#9/2: En bredde-først (bredth first) søke-algoritme gjør bruk av en stakk.

Svar: Nei Begrunnelse: Her brukes in kø.

#10/2: En dybde-først (depth-first) søke-algoritme gjør bruk av en stakk.

Svar: Ja Begrunnelse: Sist besøkte node blir undersøkt først. Dybde oppnås ved dette.

#11/3: En maksimal-matching i en bipartitt graf kan finnes ved Lineær-Programmering.

Svar: Ja Begrunnelse: LP kan brukes, men en algoritme for flytmaksimering er mer effektiv. Sistnevnte metode er kun et spesialtilfelle av et LP-problem.

#12/3: Hvis noen kantvekter i en rettet graf G = (V,E) er negative, kan den korteste veien fra node s til node t finnes ved å bruke Dijkstra's algoritme dersom vi først legger til en stor konstant C til alle E's kantlengder slik at alle disse blir ikke-negative.

Svar: Nei Begrunnelse: Moteksempel: For en graf {s,v,t}, med kanterlengder {(s,v), (v,t),(s,t)} = {-2,-1,-2} er korteste vei {s,v,t}. Med C=2 vil korteste vei bli {s,t} direkte.

#13/1: Enhver DAG (directed acyclic graph) kan på en entydig måte sorteres topologisk.

Svar: Nei Begrunnelse: Trivielt å finne et moteksempel med kun 3 noder.

#14/1: Dijkstras algoritme er et eksempel på en grådighetsalgoritme.

Svar: Ja Begrunnelse: Den inkluderer i "grådig" rekkefølge den noden /en av de noder som
ligger nærmest de noder som allerede er nådd.

#15/1: Dijkstras algorime er et eksempel på dynamisk programmering.

Svar: Ja Begrunnelse: Hittil korteste vei til en node lagres i en tabell der verdiene underveis stadig blir lavere.

#16/3: Hvis alle kant-kapasiteter til en flyt-graf er et multiplum av 5, da vil også den maksimale flyten være dette.

Svar: Ja Begrunnelse: Den maksimale flyten er lik en sum av linjekapasiteter som alle er delelig med 5. Summen, og derved flyten, blir da også delelig med 5.

#17/4: I en flyt-graf med noder s,a,b,c,d,t lar vi f/m bety at f enheter flyter på en kant med kapasitet m. Vi har gitt følgende kantflyt: (s,a):2/4, (s,c):6/6, (c,a):3/7, (a,b):5/5, (b,c):0/2, (c,d):3/3, (b,d):1/4, (d,t):4/4, (b,t):4/6. Denne kantflyten er maksimal.

Svar: Ja Begrunnelse: Flyten gir 8 enheter fra s til t, og over snittet S: {s,a,c} / {b,d,t} flyter maksimal kapasitet 8. Flyten over alle kanter er gyldig og er derved også optimal.

#18/3: La P være den korteste veien fra s til t i en graf G=(V,E). Dersom vi øker lengden på alle kanter i E med 1, så vil P fortsatt være den korteste veien fra s til t.

Svar: Nei Begrunnelse: Moteksempel: La G=(V,E) bestå av V={a,b,c} E={(a,b),(b,c), (a,c)} med vekter w(a,b) = 1, w(b,c) = 1 og w(a,c) = 2.5. Korteste vei fra a til c er da a-b-c, mens ny korteste vei blir a-c etter at vektene er øket med 1.

#19/3: Et dybde-først-søk i en graf er asymptotisk raskere enn et bredde-først-søk.

Svar: Nei Begrunnelse: Begge kjører på tid Θ (V+E)

#20/2: n heltall i området $[0,n^{100}]$ kan sorteres i lineær tid.

Svar: Ja Begrunnelse: En kan f.eks. bruke Radix-sortering.

#21/2: Grafen G er asyklisk hvis det ikke oppstår ”back-edges” under dybde-først-traversering av G.

Svar: Ja Begrunnelse: ”back-edges” signaliserer sykler. Hvis slike ikke finnes finnes heller ingen sykler.

#22/4: Vi skal flettesortere (merge) k sorterte lister, hver med n/k elementer, ved følgende metode:
Flett de 2 første listene, flett så resultatet med den tredje listen, flett dette resultatet videre med den fjerde listen, og så videre, inntil den siste listen på n/k elementer blir flettet inn.
Påstanden er her at denne algoritmen krever Θ (kn) tid.

Svar: Ja Begrunnelse:
Tid = 2n/k + 3n/k + ...+ kn/k = ...= (n/k)(k+2)(k-1)/2 = Θ (kn)

Tilstrekkelig med mellomregninger må her inngå i begrunnelsen, minst som ovenfor.