INSTITUT FOR DATALOGI, AARHUS UNIVERSITET

Science and Technology
EKSAMEN
Algoritmer og Datastrukturer 2 (2003-ordning)
Antal sider i opgavesættet (incl. forsiden): 11 (elleve)
Eksamensdag: Fredag den 14. august 2015, kl. 9.00-13.00
Tilladte medbragte hjælpemidler: Alle sædvanlige hjælpemidler (lærebøger, notater, lommeregner). Computer må ikke medbringes.
Materiale der udleveres til eksaminanden:

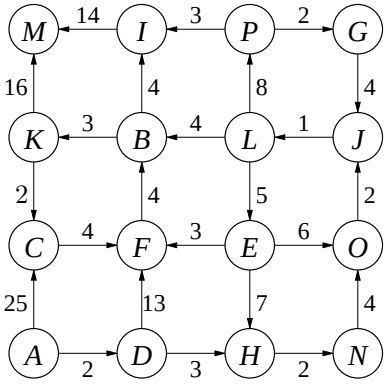
OPGAVETEKSTEN
BEGYNDER
PÅ NÆSTE SIDE

—oOo—

Opgave 1 (25%)

Bemærk: *Bagerst i eksamenssættet findes sider til at angive svarene til opgave 1 på og som afleveres som del af eksamensbesvarelsen.*

Betragt nedenstående vægtede orienterede graf. Det antages, at grafen er givet ved *incidenslister*, hvor incidenslisterne er sorteret alfabetisk.

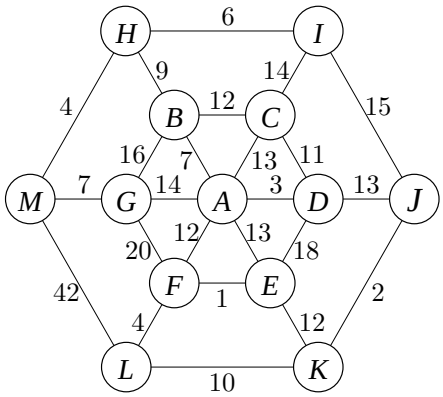


Spørgsmål a: Angiv et BFS-træ for ovenstående graf, hvor BFS-gennemløbet starter i **knuden** A. Angiv kanterne i BFS-træet, BFS-numrene for knuderne, og rækkefølgen knuderne bliver indsat i køen Q i BFS-algoritmen. □

Spørgsmål b: Angiv et DFS-træ for ovenstående graf, hvor DFS-gennemløbet starter i **knuden** A, og en DFS-nummerering af knuderne. Angiv for hver knude "discovery time" og "finishing time". Marker for hver kant om det er en "tree edge" (T), "back edge" (B), "cross edge" (C) eller "forward edge" (F). □

Spørgsmål c: Angiv de stærke sammenhængskomponenter i ovenstående graf. For hver komponent angiv knuderne i komponenten. □

Spørgsmål d: Angiv et korteste veje (SSSP) træ for ovenstående graf, når korteste veje beregningen sker med hensyn til **startknuden** A. For hver knude v angiv også afstanden fra startknuden A til v, og angiv rækkefølgen knuderne tages ud af prioritetskøen Q i Dijkstra's algoritme. □

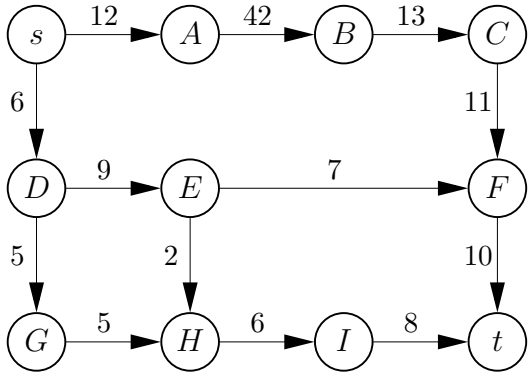


Spørgsmål e: Angiv kanterne i et minimum udspændende træ (MST) for ovenstående uorienterede graf med vægte på kanterne. Angiv i hvilken rækkefølge knuderne bliver taget ud af prioritetskøen Q i Prim's algoritme, når denne starter i **knuden** A. □

Opgave 2 (15%)

Bemærk: *Bagerst i eksamenssættet findes en side til at angive svarene til opgave 2 på og som afleveres som del af eksamensbesvarelsen.*

Betragt nedenstående netværk med de angivne kapaciteter på kanterne.



Spørgsmål a: Angiv en maksimal strømning (maximum flow) fra s til t i netværket (angiv for hver kant strømningen langs kanten), angiv værdien af en maksimal strømning, og angiv et snit (cut) (dvs. opdeling af knuderne i to disjunkte mængder) hvor kapaciteten af snittet er lig værdien af en maksimal strømning. □

Spørgsmål b: Betragt Edmonds-Karp algoritmen anvendt på ovenstående graf til beregning af en maksimal strømning. Angiv de forbedrende stier (augmenting paths) der anvendes under udførslen af Edmonds-Karp algoritmen. For hver forbedrende sti angiv knuderne på stien og strømningen, man forbedrer med, langs stien. □

Opgave 3 (20%)

Betragt en orienteret graf $G = (V, E)$ med $n = |V|$ knuder og $m = |E|$ kanter, og hvor hver kant $e \in E$ har en ikke negativ heltallig længde $w(e)$. Længden af en orienteret sti i grafen er summen af længderne af kanterne på stien, og afstanden $d(u, v)$ mellem to knuder $u \in V$ og $v \in V$ er længden af en korteste sti mellem u og v i grafen.

Lad $U \subset V$ være en mængde af $k = |U|$ *vigtige* knuder og lad $s \in V \setminus U$ være en *startknude*, der ikke er en vigtig knude.

Vi ønsker at *besøge* vigtig knuder. En vigtig knude v kan besøges hvis og kun hvis der findes orienterede stier fra s til v *og* fra v til s. *Omkostningen* for at besøge v er $\delta_s(v) = d(s, v) + d(v, s)$ (hvor $\delta_s(v) = +\infty$ hvis det ikke er muligt at besøge v).

Spørgsmål a: Beskriv en algoritme der bestemmer hvilke vigtige knuder det er muligt at besøge fra startknuden s. Angiv algoritmens udførselstid som funktion af n, m og k. En ideel løsning opnår udførselstid $O(n + m)$. □

Spørgsmål b: Beskriv en algoritme, der givet et positivt heltal Δ, bestemmer hvilke vigtige knuder det er muligt at besøge fra s med omkostning højest Δ, dvs. finde alle knuder v med $\delta_s(v) \leq \Delta$. Angiv algoritmens udførselstid som funktion af n, m og k. □

Spørgsmål c: Beskriv en algoritme der finder en maksimal delmængde $W \subseteq U$ af vigtige knuder, således at $|W|$ er størst mulig og den samlede omkostning for at besøge disse vigtige knuder er højest Δ, dvs. $\sum_{v \in W} \delta_s(v) \leq \Delta$. Bemærk at man skal tilbage til startknuden imellem hvert besøg af en vigtig knude. Angiv algoritmens udførselstid som funktion af n, m og k. □

I det foregående antog vi at startknuden var givet. I næste spørgsmål ønsker vi at finde en startknude s hvor antallet af vigtige knuder man kan besøge med total omkostning Δ er størst mulig.

Spørgsmål d: Beskriv en algoritme, der givet et positivt heltal Δ, finder en startknude s, således at det maksimale antal vigtige knuder man kan besøge inden for total omkostning Δ er størst mulig, dvs. find et s således at der findes en størst mulig mængde vigtige knuder $W \subseteq U$ hvor $\sum_{v \in W} \delta_s(v) \leq \Delta$. Angiv algoritmens udførselstid som funktion af n, m og k. □

Opgave 4 (25%)

I denne opgave har vi givet et positivt heltal n og k mønter med positive og heltallige værdier $c_1 \leq c_2 \leq \cdots \leq c_k$. Vi ønsker at afgøre om vi kan opnå n som summen af en delmængde af mønterne, hvor hver mønt må indgå 0 eller 1 gang. Bemærk der kan være flere mønter med samme værdi.

F.eks. for de fem mønter 2, 2, 3, 5 og 11 kan vi opnå værdien $15 = 2+2+11$, hvorimod det ikke er muligt at opnå summen 17 med disse mønter.

For $j \geq 0$ og $i \geq 0$, lader vi $B(j,i)$ angive om det er muligt at opnå i som summen af en delmængde af mønterne $c_1, c_2, \ldots, c_j$.

$B(j,i)$ kan bestemmes ved følgende rekursionsformel.

$$B(j,i) = \begin{cases} \text{sand} & \text{hvis } i = 0 \\ \text{falsk} & \text{hvis } i > 0 \wedge j = 0 \\ B(j-1,i) & \text{hvis } i > 0 \wedge j > 0 \wedge c_j > i \\ B(j-1,i) \vee B(j-1,i-c_j) & \text{hvis } i > 0 \wedge j > 0 \wedge c_j \leq i \end{cases}$$

Spørgsmål a: Udfyld nedenstående tabel for mønterne $c_1 = c_2 = 2$, $c_3 = 3$ og $c_4 = 5$ (det er tilstrækkeligt at angive sande indgange med "T").

$B(j,i)$	0	1	2	3	4	5	6	7	8	9	10	11	12
0													
1													
2													
3													
4													

□

Spørgsmål b: Angiv en algoritme baseret på dynamisk programmering, der givet en mængde positive heltallige mønter $c_1, \ldots, c_k$, og et positivt heltal n, afgør om n kan opnås som summen af en delmængde af mønterne. Angiv algoritmens udførselstid. □

Spørgsmål c: Udvid algoritmen fra spørgsmål b) til at udskrive en delmængde af mønterne med sum n, hvis en sådan løsning findes. Angiv algoritmens udførselstid. □

Hvis n kan skrives som en sum af en delmængde af mønterne og alle løsninger bruger det samme antal mønter og med de samme værdier, så siger vi at løsningen er *entydig*.

F.eks. for $c_1 = 1$, $c_2 = c_3 = 3$ og $c_4 = 5$, så har $n = 8$ en entydig løsning da $8 = 3 + 5$ $(= c_2 + c_4 = c_3 + c_4)$, hvorimod $n = 6$ ikke har en entydig løsning da $6 = 1 + 5 = 3 + 3$ $(= c_1 + c_4 = c_2 + c_3)$.

Spørgsmål d: Beskriv en algoritme baseret på dynamisk programmering der afgør om n kan skrives som summen af en delmængde af mønterne $c_1, \ldots, c_k$, og hvis der findes en løsning, om løsningen er entydig. Angiv algoritmens udførselstid. □

Opgave 5 (15%)

For en streng $T[1..n]$ af længde n over et alfabet med $O(1)$ tegn, og et positivt heltal k, ønsker vi at finde et par af positioner i og j, hvor $i < j$, således at $T[i..i+k-1] = T[j..j+k-1]$ og at $j - i$ er mindst mulig. Dvs. vi ønsker at finde en delstreng af T af længde k som forekommer to gange tæt på hinanden. Det kan antages at der altid findes en delstreng af længde k som forekommer mindst to gange i T.

For $T =$ `abcabbabacccaba`$\overset{i}{\underline{\texttt{abb}}}$`cb`$\overset{j}{\underline{\texttt{abb}}}$`acbaabab` og $k = 3$, har vi f.eks. $i = 16$ og $j = 21$.

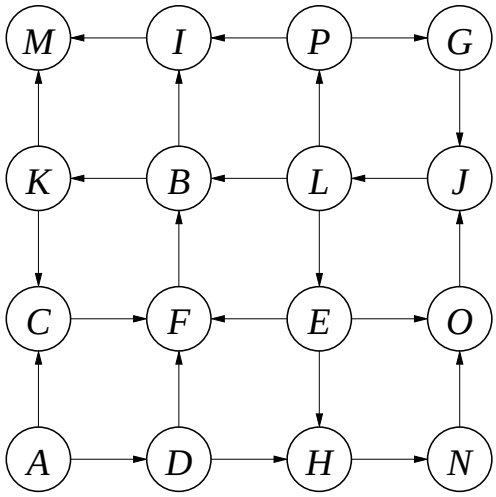
I det følgende kan det antages at et suffikstræ for en streng af længde n over et alfabet med $O(1)$ tegn kan konstrueres i $O(n)$ tid.

Spørgsmål a: Beskriv en algoritme der givet en streng T af længde n og et heltal k, finder et par af positioner $i < j$, således at $T[i..i+k-1] = T[j..j+k-1]$ og $j-i$ er mindst mulig. Angiv algoritmens udførselstid. En ideel løsning opnår udførselstid $O(n)$. For at få fuld point for opgaven skal udførselstiden for en korrekt løsning være $O(n \log n)$. □

Årskort:	Navn:

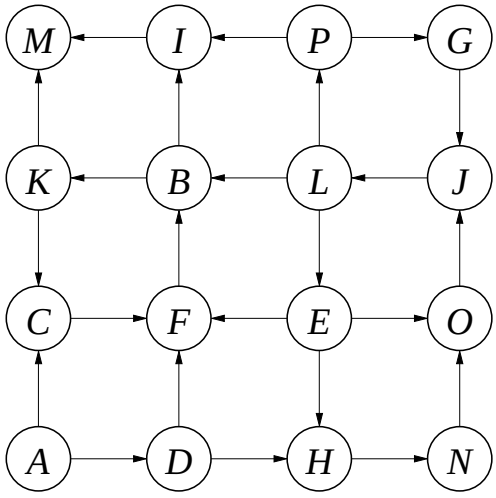
Opgave 1 — Svar

Spørgsmål a: BFS



Indsættelser i Q: _____

Spørgsmål b: DFS



(blank side)

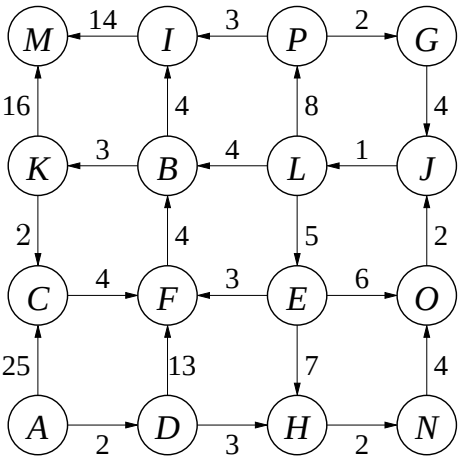
Årskort:	Navn:

Opgave 1 — Svar

Spørgsmål c:

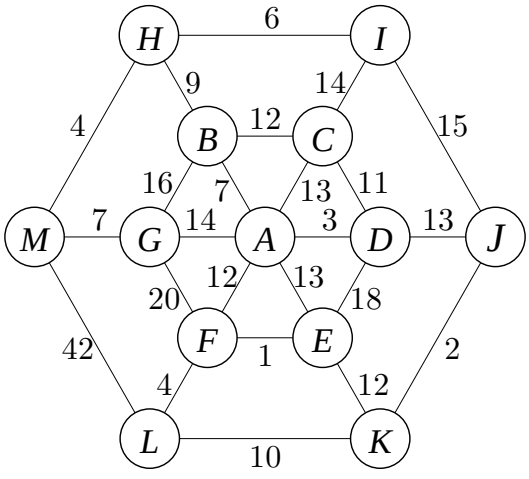
Stærke sammenhængskomponenter: _____

Spørgsmål d: SSSP



Udtagelse fra Q: _____

Spørgsmål e: MST



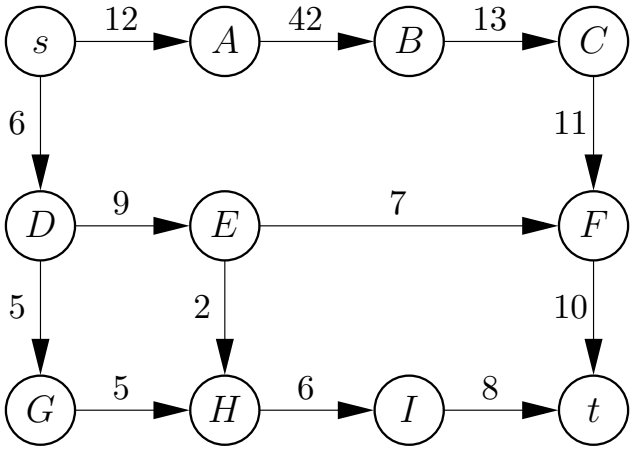
Udtagelse fra Q: _____

(blank side)

Årskort:	Navn:

Opgave 2 — Svar

Spørgsmål a:



Værdien af strømning: _____

Minimal snit: _____

Spørgsmål b:

Forbedring	Forbedrende sti