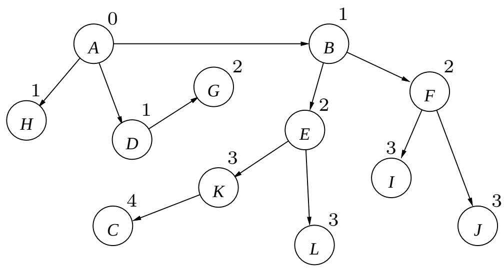
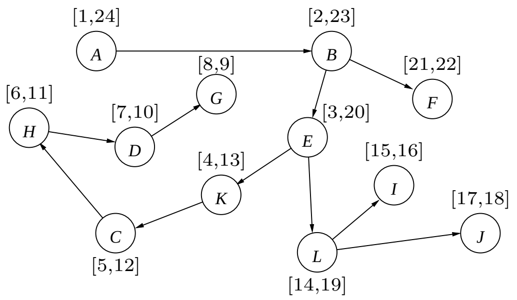
1a

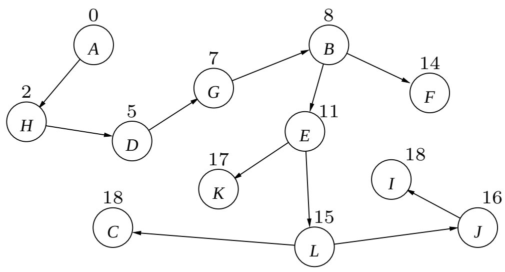


Indsættelser i Q: $ABDHEFGKLIJC$

1b



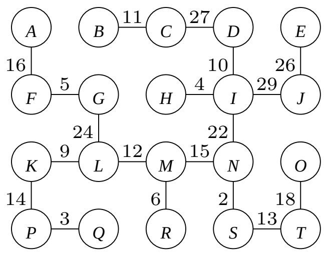
1c



1d

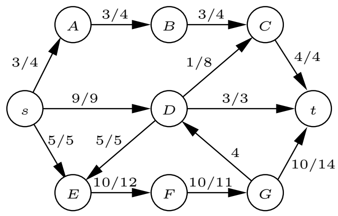
$\{A\}, \{F\}, \{I\}, \{J\}, \{B, C, D, E, G, H, K, L\}$

1e



Fjernelser fra Q: $AFGLKMRPQNSTOIHDCBJE$

2a



Maximal strømning = 17.
Snit med kapacitet 17: $(\{s, A, B, C, D\}, \{E, F, G, t\})$

2b

Forbedring	Sti
3	sDt
4	$sDCt$
5	$sEFGt$
2	$sDEFGt$
3	$sABCDEFGt$

3a

Lav en uorienteret graf $G = (V, E)$, hvor der for hver ikke-blokkeret celle er en knude og hvor der er uorienterede kanter mellem to ikke-blokkerede naboceller. $|V| \leq mn$ og $|E| \leq m(n-1) + (m-1)n$. Udfør BFS startende i knude $(1, i)$. Der findes en sti af længde $\leq t$ hvis og kun hvis der mindst findes én knude i øverste række med BFS nummer $\leq t$. Tid $O(mn)$.

3b

Lav en orienteret graf $G = (V, E)$, hvor der for hver ikke-blokkerede celle (i, j) og tid k, $0 \leq k \leq t$, findes en knude $(i, j, k) \in V$, og hvor der mellem to ikke-blokkerede naboceller (i, j) og (i', j'), findes kanter $((i, j, k), (i', j', k+1)) \in E$, for $0 \leq k < t$, og for hver ikke-blokkerede celle (i, j) findes kanter $((i, j, k), (i, j, k+1)) \in E$, for $0 \leq k < t$. Desuden tilføjes en knude S og en knude T, og kanter fra S til alle knuder $(1, j, 0)$, hvor $(1, j) \in I_0$, og kanter fra (m, j, t) til T, hvor $1 \leq j \leq n$. Vi ønsker nu at finde det maksimale antal knude-disjunkte stier i denne graf. De knuder $(1, j, 0)$ der indgår i disse stier returneres. Vi har at $|V| \leq mn(t+1) + 2$ og $|E| \leq k + n + 5nmt$. Stierne findes ved at give alle knuderne, på nær S og T, en kapacitet på 1 og så køre en max-flow algoritme, f.eks. Ford-Fulkerson i tid $O(k|E|) = O(knmt)$.

4a

$C(i,j)$	1	2	3	4
0	0	0	0	0
1	0	0	0	0
2	2	0	0	0
3	3	1	0	0
4	5	2	1	0
5	7	3	2	1
6	8	3	2	1
7	9	4	2	2

4b

```
for i=0 to n
  for j=1 to k
    if i=0 then L[i,j]=0
    else if j=1 then
      L[i,j] = x[i] - x[1]
    else
      L[i,j] = +infinity
      for t=1 to i
        v = max(L[t-1,j-1],x[i]-x[t])
        if v<L[i,j] then
          L[i,j] = v
          B[i,j] = t  (* til spørgsmål c) *)
return L[n,k]
```

Tid $O(n^2k)$.

4c

```
code from 4b)
report(n,k)

proc report(i,j)
  if i>0 then
    if j=1 then output x[1]..x[i]
    else
      t = B[i,j]
      report(t-1,j-1)
      output x[t]..x[i]
```

Tid $O(n^2k)$.

5a

$yXy =$ `cabdc`, som forekommer på positionerne 3, 7, 15 og 21.

5b

Løsning 1: For hver position i, $1 \leq i \leq n$, find den mindste position j, $i < j \leq n$, hvor $S[i] = S[j]$. Hvis sådan et j findes, kør KMP algoritmen på S med mønsteret $S[i..j]$. Rapporter en delstreng $S[i..j]$, hvor KMP fandt det største antal forekomster. Tid $O(n^2)$, da der bruges $O(n)$ tid for hvert i til at finde det tilhørende j, og for at køre KMP algoritmen.

Løsning 2: Konstruer suffix træet T for $S\$$. For hvert barn v af roden af T, find det første tegn y på kanten fra roden til v. Hvis y findes yderlige på en kant ned til en knude w i undertræet indeholdende v (incl. kanten fra roden til v), så udgør stien fra roden til denne anden forekomst af y en delstreng yXy, hvor antal forekomster af yXy i S er antal blade i undertræet rodet i w. Størrelserne af undetræerne kan beregnes i $O(n)$ tid ved et DFS gennemløb af T, men checket for om y indgår i en kant tager tid proportionalt med antal symboler på kanten som er $\leq n$. Totalt skal højst n kanter undersøges, hvilket giver $O(n^2)$ tid. (Ved brug af radix sort kan alle n tests udføres i $O(n)$ tid.)