EKSAMEN

Algoritmer og Datastrukturer

Fredag den 29. maj 2020, kl. 9.00–11.00

Institut for Datalogi, Naturvidenskabelige Fakultet, Aarhus Universitet

Tilladte hjælpemidler: **Alle**

Kommunikation med andre om eksamensopgaverne er ikke tilladt under eksamen

Studienummer : ____________________

Navn : ____

I nødstilfælde ring 50 59 54 32 (Gerth Stølting Brodal)

Vejledning og pointgivning

Dette eksamenssæt består af en mængde multiple-choice-opgaver.

Opgaverne besvares på opgaveformuleringen **som afleveres**.

For hver opgave er angivet opgavens andel af det samlede eksamenssæt.

Hvert delspørgsmål har præcist ét rigtigt svar.

For hvert delspørgsmål må du vælge **max ét svar** ved at afkrydse den tilsvarende rubrik.

Et delspørgsmål bedømmes som følgende:

- Hvis du sætter kryds ved det rigtige svar, får du 1 point.
- Hvis du ikke sætter nogen krydser, får du 0 point.
- Hvis du sætter kryds ved et forkert svar, får du $-\frac{1}{k-1}$ point, hvor k er antal svarmuligheder.

For en opgave med vægt $v\,\%$ og med n delspørgsmål, hvor du opnår samlet s point, beregnes din besvarelse af opgaven som:

$$\frac{s}{n} \cdot v\,\%$$

Bemærk at det er muligt at få negative point for en opgave.

Opgave 1 (Asymptotisk notation, 6 %)

I det følgende angiver $\log n$ 2-tals-logaritmen af n.

	Ja	Nej
n^5 er $O(3^n)$	☒	[B]
$n^3/\log n$ er $O(n^2)$	[A]	☒
$n^2/2$ er $O(\log n)$	[A]	☒
3^3 er $O(\sqrt{n})$	☒	[B]
$2^{3\log n}$ er $O(3^n)$	☒	[B]
n^2 er $O(n \cdot \log n)$	[A]	☒
$n \cdot \log n + (\log n)^3$ er $O(\sqrt{n} \cdot \log n)$	[A]	☒
$5 \cdot 2^{\log n} + \sqrt{n}$ er $O(n^{3/2})$	☒	[B]
$n^{0.1}/7 + n^{3/2}$ er $O(n^{2/3})$	[A]	☒
$(\log n)^2 + 2^{3\log n}$ er $\Omega(n!)$	[A]	☒
$\sqrt{n}$ er $\Theta(2^n)$	[A]	☒
$\log(n!)$ er $\Omega(n^2)$	[A]	☒

Opgave 2 (Analyse af løkker, 6 %)

```
Algoritme loop1(n)
  s = 1
  for i = 1 to n
    for j = 1 to n + 1 − i
      s = s + 1
```

```
Algoritme loop2(n)
  s = 1
  for i = 1 to n
    for j = i to 2 ∗ i
      s = s + 1
```

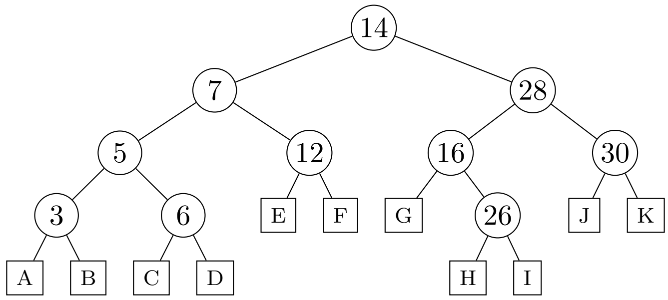
```
Algoritme loop3(n)
  i = 0
  j = 0
  while i < n
    if j < n
      j = j + 1
    else
      j = 0
      i = i + 1
```

```
Algoritme loop4(n)
  i = 1
  while i ≤ n ∗ n
    j = 1
    k = 1
    while k ≤ i
      k = k + j
      j = j + 1
    i = 2 ∗ i
```

Angiv for hver af ovenstående algoritmer udførselstiden som funktion af n i Θ-notation.

	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(\sqrt{n})$	$\Theta(\log n)$	$\Theta(n^2)$	$\Theta(\sqrt{n}\log n)$	$\Theta(n^3)$	$\Theta((\log n)^2)$
loop1	[A]	[B]	[C]	[D]	☒	[F]	[G]	[H]
loop2	[A]	[B]	[C]	[D]	☒	[F]	[G]	[H]
loop3	[A]	[B]	[C]	[D]	☒	[F]	[G]	[H]
loop4	[A]	☒	[C]	[D]	[E]	[F]	[G]	[H]

Opgave 3 (Indsættelser i søgetræer, 4 %)



Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 21, 13, 27, 4 og 19 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

	A	B	C	D	E	F	G	H	I	J	K
Insert(21)	A	B	C	D	E	F	G	☒ H	I	J	K
Insert(13)	A	B	C	D	E	☒ F	G	H	I	J	K
Insert(27)	A	B	C	D	E	F	G	H	☒ I	J	K
Insert(4)	A	☒ B	C	D	E	F	G	H	I	J	K
Insert(19)	A	B	C	D	E	F	G	☒ H	I	J	K

Opgave 4 (Max-Heap-Insert, 4 %)

Angiv den binære max-heap efter indsættelse af elementerne 13, 6, 11, 10, 12, 14 og 3 i den givne rækkefølge med Max-Heap-Insert, startende med den tomme heap.

1	2	3	4	5	6	7	
13	6	11	10	12	14	3	A

1	2	3	4	5	6	7	
13	12	14	10	6	11	3	B

1	2	3	4	5	6	7	
14	13	12	11	10	6	3	C

1	2	3	4	5	6	7	
14	12	13	6	10	11	3	☒ D

1	2	3	4	5	6	7	
14	12	13	10	6	11	3	E

Opgave 5 (Build-Max-Heap, 4 %)

1	2	3	4	5	6	7	8	9
9	3	5	1	4	7	8	6	2

Hvad er resultat af BUILD-MAX-HEAP på ovenstående array ?

1	2	3	4	5	6	7	8	9	
9	6	8	3	4	7	5	1	2	☒

1	2	3	4	5	6	7	8	9	
9	6	8	4	3	5	7	1	2	B

1	2	3	4	5	6	7	8	9	
1	2	3	4	5	6	7	8	9	C

1	2	3	4	5	6	7	8	9	
9	8	7	6	5	4	3	2	1	D

1	2	3	4	5	6	7	8	9	
9	4	8	6	3	7	5	1	2	E

Opgave 6 (Heap-Extract-Max, 4 %)

1	2	3	4	5	6	7	8	9	10	11	12	13
26	21	17	12	19	16	2	1	9	8	11	13	15

Hvad er resultat af HEAP-EXTRACT-MAX på ovenstående max-heap ?

1	2	3	4	5	6	7	8	9	10	11	12	
21	19	17	12	15	16	2	1	9	8	11	13	☒

1	2	3	4	5	6	7	8	9	10	11	12	
21	19	17	12	11	16	2	1	9	8	15	13	B

1	2	3	4	5	6	7	8	9	10	11	12	
21	19	17	12	11	16	2	1	9	8	13	15	C

1	2	3	4	5	6	7	8	9	10	11	12	13	
21	19	17	12	11	16	2	1	9	8		13	15	D

1	2	3	4	5	6	7	8	9	10	11	12	
21	19	15	17	16	12	1	9	8	11	13	2	E

Opgave 7 (Partition, 4 %)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
10	20	16	3	30	6	21	24	7	2	26	29	9	14	22

Angiv resultatet af at anvende PARTITION$(A, 2, 13)$ på ovenstående array A.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	10	2	3	6	7	9	16	20	21	24	26	29	30	14	22
B	10	3	6	7	2	9	20	16	30	21	24	26	29	14	22
☒ C	10	3	6	7	2	9	21	24	20	30	26	29	16	14	22
D	2	3	6	7	9	10	14	16	20	21	22	24	26	29	30

Opgave 8 (Lineær probing, 4 %)

0	1	2	3	4	5	6	7	8	9	10
11	0	12					9			16

I ovenstående hashtabel af størrelse 11 er anvendt *linear probing* med hashfunktionen $h(k) = 2k \bmod 11$.

Angiv positionerne de fem elementer 1, 5, 6, 8 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 9, 11, 12 og 16).

	0	1	2	3	4	5	6	7	8	9	10
INSERT(1)	A	B	C	☒ D	E	F	G	H	I	J	K
INSERT(5)	A	B	C	☒ D	E	F	G	H	I	J	K
INSERT(6)	A	B	C	☒ D	E	F	G	H	I	J	K
INSERT(8)	A	B	C	D	E	☒ F	G	H	I	J	K
INSERT(10)	A	B	C	D	E	F	G	H	I	☒ J	K

Opgave 9 (Dobbelt hashing, 4 %)

0	1	2	3	4	5	6	7	8	9	10
11	0				19	17	20			

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 2k \bmod 11$ og $h_2(k) = 1 + (2k \bmod 10)$.

Angiv positionerne de fem elementer 2, 3, 5, 6 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 11, 17, 19 og 20).

	0	1	2	3	4	5	6	7	8	9	10
INSERT(2)	A	B	C	D	☒	F	G	H	I	J	K
INSERT(3)	A	B	☒	D	E	F	G	H	I	J	K
INSERT(5)	A	B	C	D	E	F	G	H	I	J	☒
INSERT(6)	A	B	C	D	☒	F	G	H	I	J	K
INSERT(9)	A	B	C	☒	E	F	G	H	I	J	K

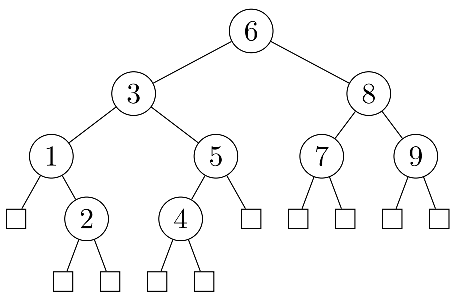
Opgave 10 (Rekursionsligninger, 4 %)

Angiv løsningen for hver af nedenstående rekursionsligninger, hvor $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$
$T(n) = 2 \cdot T(n/5) + n$	A	B	☒	D	E	F	G
$T(n) = 3 \cdot T(n/9) + 1$	A	☒	C	D	E	F	G
$T(n) = 4 \cdot T(n/2) + n^2$	A	B	C	D	E	☒	G
$T(n) = T(n-1) + \log n$	A	B	C	☒	E	F	G
$T(n) = T(n-1) + n^2$	A	B	C	D	E	F	☒

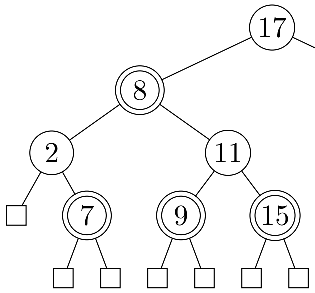
Opgave 11 (Rød-sort træ, 4 %)

For hver af nedenstående delmængder, angiv om nedenstående binære træ er et lovligt rød-sort træ hvis netop disse knuder farves røde.



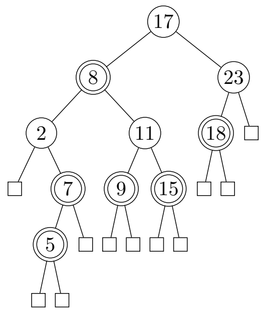
	Ja	Nej
2, 3, 4, 8	☒	B
2, 4, 6	A	☒
2, 3, 4, 7, 9	☒	B
1, 2, 4, 5, 7, 9	A	☒
2, 4	☒	B

Opgave 12 (Indsættelse i rød-sort træer, 4 %)

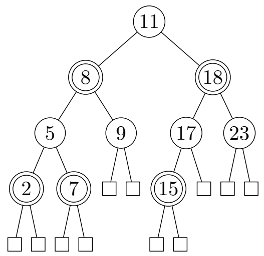


Angiv det resulterende rød-sorte træ når man indsætter 5 i ovenstående rød-sorte træ (dobbeltcirkler angiver røde knuder).

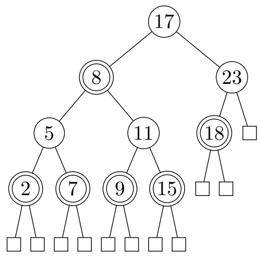
A



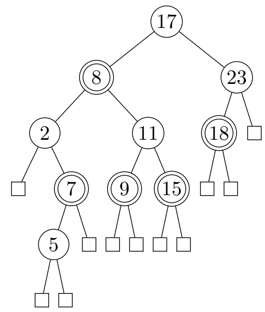
B



$\boxtimes$



D

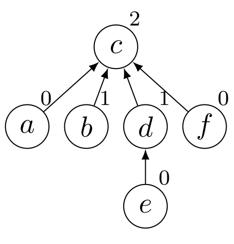


Opgave 13 (Union-find, 4 %)

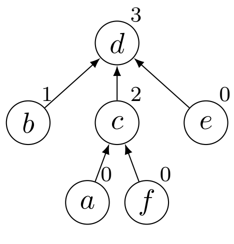
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

MAKESET(a)
MAKESET(b)
MAKESET(c)
MAKESET(d)
MAKESET(e)
MAKESET(f)
UNION(a, b)
UNION(f, c)
UNION(a, f)
UNION(e, d)
UNION(a, e)
FIND-SET(b)

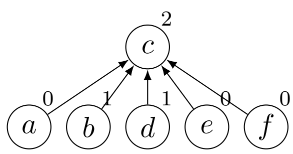
$\boxtimes$



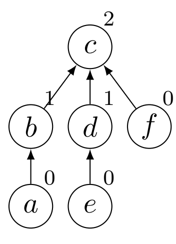
B



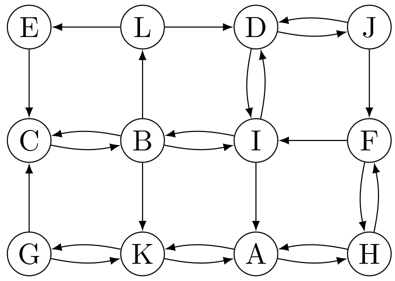
C



D



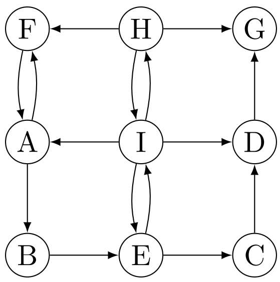
Opgave 14 (BFS, 4 %)



For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver indsat i køen Q i BFS-algoritmen. Det antages, at grafen er givet ved incidenslister, hvor incidenslisterne er sorteret alfabetisk.

A	B	☒	D
A H K F G I C D B J L E	A H F I B C K G L D J E	A H K F G I C B D L J E	A K H G F C I B D L J E

Opgave 15 (DFS, 4 %)



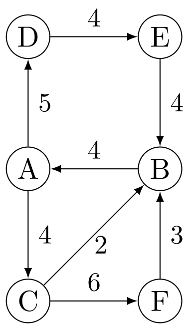
Betragt et dybde først gennemløb (DFS) af ovenstående graf, hvor DFS-gennemløbet starter i **knuden A**, hvor de udgående kanter til en knude besøges i alfabetisk rækkefølge. Angiv i hvilken rækkefølge knuderne får tildelt **finishing time**.

☒	B	C	D
G D C F H I E B A	F G H D I C E B A	G H D I C E F B A	F H I G D C E B A

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet.

	Tree edge	Back edge	Cross edge	Forward edge
(I, D)	A	B	☒	D
(H, F)	☒	B	C	D
(I, E)	A	☒	C	D
(A, F)	A	B	C	☒

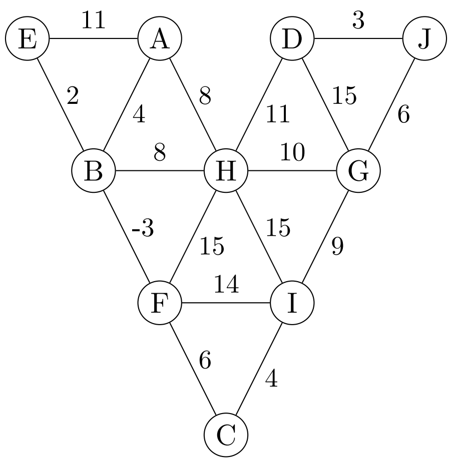
Opgave 16 (Dijkstras algoritme, 4 %)



Antag Dijkstras algoritme anvendes til at finde korteste afstande fra **knuden A** til alle knuder i ovenstående graf. Angiv hvilken rækkefølge knuderne bliver taget ud af prioritetskøen i Dijkstra's algoritme.

A	☒	C	D
A C D B F E	A C D B E F	A C B F D E	A C D E B F

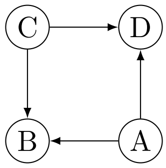
Opgave 17 (Prims algoritme, 4 %)



Antag Prims algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf, og algoritmen starter i **knuden A**. Angiv hvilken rækkefølge knuderne bliver inkluderet i det minimum udspændende træ (taget ud af prioritetskøen i Prims algoritme).

A	B	☒	D
A B F E C H I G J D	A B F E C H I G D J	A B F E C I H G J D	A B F C I G J D H E

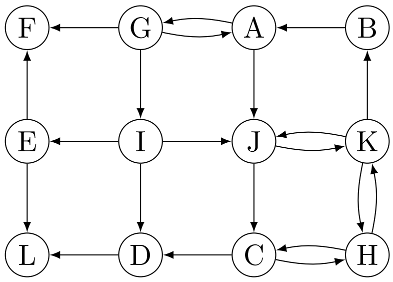
Opgave 18 (Topologisk sortering, 4 %)



Angiv for hver af nedenstående ordninger af knuderne i ovenstående graf om det er en lovlig topologisk sortering.

	Ja	Nej
A C B D	☒	B
C B D A	A	☒
D A B C	A	☒
C A B D	☒	B
C A D B	☒	B

Opgave 19 (Stærke sammenhængskomponenter, 4 %)



Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

A	B	C	D	☒	F	G	H	I	J	K	L
1	2	3	4	5	6	7	8	9	10	11	12

Opgave 20 (Invariant, 4 %)

Givet to ikke-negative heltal n og m, så beregner nedenstående algoritme $n \cdot m$.

Algoritme Multiplikation(n)
Inputbetingelse : Heltal $n \geq 0$ og $m \geq 0$
Outputkrav : $r = n_0 \cdot m_0$
Metode : $r \leftarrow 0$
  $\{I\}$ **while** $n > 0$ **do**
    if n er ulige **then**
      $r \leftarrow r + m$
      $n \leftarrow n - 1$
    else
      $m \leftarrow m * 2$
      $n \leftarrow n/2$

For hvert af nedenstående udsagn, angiv om de er en invariant I for algoritmen Multiplikation, hvor n_0 og m_0 angiver værdierne for henholdsvis n og m i starten.

	Ja	Nej
$0 \leq n \leq n_0$	☒	B
$0 \leq m \leq m_0$	A	☒
$r = m \cdot n$	A	☒
$n_0 \cdot m_0 = r + n \cdot m$	☒	B
$n_0 \cdot m_0 + r = n \cdot m$	A	☒

Dynamisk programmering

De næste fire opgaver vedrører at løse LCS^2 problemet ved hjælp af dynamisk programmering.

Vi antager at vi har givet to strenge $A{=}a_0a_1\dots a_{n-1}$ og $B{=}b_0b_1\dots b_{m-1}$ af længde henholdsvis n og m. En fælles delsekvens er en sekvens af tegn der både er en delsekvens af A og B. F.eks. er $\mathtt{a\,b\,a\,b\,b}$ en fælles delsekvens for $A{=}\underline{\mathtt{a}}\,\underline{\mathtt{b}}\,\underline{\mathtt{a}}\,\underline{\mathtt{b}}\,\mathtt{c}\,\underline{\mathtt{b}}$ og $B{=}\underline{\mathtt{a}}\,\underline{\mathtt{b}}\,\mathtt{c}\,\underline{\mathtt{a}}\,\underline{\mathtt{b}}\,\underline{\mathtt{b}}$. En anden fælles delsekvens for $A{=}\mathtt{a\,b}\,\underline{\mathtt{a\,b\,c}}\,\underline{\mathtt{b}}$ og $B{=}\underline{\mathtt{a\,b\,c}}\,\mathtt{a}\,\underline{\mathtt{b}}\,\mathtt{b}$ består af to *sammenhængde blokke* $\mathtt{a\,b\,c}$ og b, af længde henholdsvis 3 og 1, fra både A og B. I LCS^2 problemet ønsker vi at finde en fælles delsekvens hvor *summen af kvadraterne af blok-længderne* er størst mulig, også betegnet LCS^2 scoren. I dette eksempel er $3^2 + 1^2 = 10$ den maksimale LCS^2 score. Bemærk at en fælles delsekvens, der opnår en maksimal LCS^2 score, er ikke nødvendigvis en længste fælles delsekvens.

Vi lader $S(i,j)$ betegne den maksimale LCS^2 score for $a_ia_{i+1}\dots a_{n-1}$ og $b_jb_{j+1}\dots b_{m-1}$.

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

$$S(i,j) = \begin{cases} 0 \quad \text{hvis } i=n \text{ eller } j=m \\ \max\left\{S(i+1,j),\ S(i,j+1),\ \max\limits_{\substack{0<k\leq\min(n-i,m-j)\\ a_i\dots a_{i+k-1}=b_j\dots b_{j+k-1}}} (S(i+k,j+k)+k^2)\right\} & \text{ellers} \end{cases}$$

De følgende 4 opgaver består i at udfylde 4 blokke i følgende algoritmeskabelon.

Algoritme LCS2(A,B)
  $n = |A|$
  $m = |B|$
  Opret tom tabel $T[0..n, 0..m]$ til $S(i,j)$
  Opret tom tabel $K[0..n, 0..m]$ til back-tracking

for ...   **for** ... $\ll$ **Opgave 21**: iterer over T $\gg$
$\ll$ **Opgave 22**: beregn $T[i,j] = S(i,j)$ $\gg$
$\ll$ **Opgave 23**: udskriv LCS^2 scoren af A og B $\gg$
$\ll$ **Opgave 24**: udskriv delsekvens med maksimal LCS^2 score $\gg$

Opgave 21 (4 %)

For hver af nedenstående stykker kode, angiv om det vil kunne føre til en korrekt løsning.

Ja ☐	Ja ☐	Ja ☒
Nej ☒	Nej ☒	Nej ☐

for $i = 0$ **to** n
     for $j = 0$ **to** m
	for $i = n$ **to** 0 **step** -1
	     for $j = 0$ **to** m
		for $i = n$ **to** 0 **step** -1
		     for $j = m$ **to** 0 **step** -1

Opgave 22 (4 %)

For hver af nedenstående stykker kode, angiv om det vil kunne føre til en korrekt løsning.

Ja [A]
Nej [X]

if $i = n$ **or** $j = m$ **then**
  $T[i, j] = 0$
else
  $T[i, j] = \max(T[i + 1, j], T[i, j + 1])$
  $k = \min(n - i, m - j)$
  while $k \geq 1$ **and** $A[i + k - 1] = B[j + k - 1]$ **do**
    if $k^2 + T[i + k, j + k] > T[i, j]$ **then**
      $T[i, j] = k^2 + T[i + k, j + k]$
      $K[i, j] = k$
    $k = k - 1$

Ja [X]
Nej [B]

if $i = n$ **or** $j = m$ **then**
  $T[i, j] = 0$
else
  $T[i, j] = \max(T[i + 1, j], T[i, j + 1])$
  $k = 1$
  while $k \leq \min(n - i, m - j)$ **and** $A[i + k - 1] = B[j + k - 1]$ **do**
    if $k^2 + T[i + k, j + k] > T[i, j]$ **then**
      $T[i, j] = k^2 + T[i + k, j + k]$
      $K[i, j] = k$
    $k = k + 1$

Ja [A]
Nej [X]

if $i = n$ **or** $j = m$ **then**
  $T[i, j] = 0$
else
  $T[i, j] = \max(T[i + 1, j], T[i, j + 1])$
  for $k = 1$ **to** $\min(n - i, m - j)$
    if $A[i + k - 1] = B[j + k - 1]$ **and** $k^2 + T[i + k, j + k] > T[i, j]$ **then**
      $T[i, j] = k^2 + T[i + k, j + k]$
      $K[i, j] = k$

Opgave 23 (4 %)

For hver af nedenstående stykker kode, angiv om det vil kunne føre til en korrekt løsning.

Ja [X]	Ja [A]	Ja [A]
Nej [B]	Nej [X]	Nej [X]
print $T[0, 0]$	**print** $T[n - 1, m - 1]$	**print** $T[n, m]$

Opgave 24 (4 %)

For hver af nedenstående stykker kode, angiv om det vil kunne føre til en korrekt løsning.

Ja [A]
Nej [☒]

$i = 0$
$j = 0$
while $i < n$ **and** $j < m$ **do**
    $k = K[i, j]$
    print $a_i a_{i+1} \dots a_{i+k-1}$
    $i = i + k$
    $j = j + k$

Ja [A]
Nej [☒]

$i = 0$
$j = 0$
while $i < n$ **and** $j < m$ **do**
    if $T[i, j] = T[i + 1, j]$ **or** $T[i, j] = T[i, j + 1]$ **then**
        $i = i + 1$
        $j = j + 1$
    else
        $k = K[i, j]$
        print $a_i a_{i+1} \dots a_{i+k-1}$
        $i = i + k$
        $j = j + k$

Ja [☒]
Nej [B]

$i = 0$
$j = 0$
while $i < n$ **and** $j < m$ **do**
    if $T[i, j] = T[i + 1, j]$ **then**
        $i = i + 1$
    else if $T[i, j] = T[i, j + 1]$ **then**
        $j = j + 1$
    else
        $k = K[i, j]$
        print $a_i a_{i+1} \dots a_{i+k-1}$
        $i = i + k$
        $j = j + k$