Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = T(3n/4) + n$$

1.f: $T(n) = \Theta(n)$

$$T(n) = 3 \cdot T(n/3) + n$$

2.e: $T(n) = \Theta(n \log n)$

$$T(n) = T(n/4) + \log n$$

3.g: Rekursionsligningen kan ikke løses med Master Theorem.

3 point

Udfør HEAP-EXTRACT-MAX(A) på nedenstående array A.

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

Hvilket af nedenstående svar angiver udseendet af A bagefter?

Selected Answer:

		1	2	3	4	5	6	7	8	9
8.d:	A:	4	4	3	3	1	2	3	2	

Answers:

		1	2	3	4	5	6	7	8	9
8.a:	A:	4	4	3	3	3	2	2	1	

		1	2	3	4	5	6	7	8	9
8.b:	A:		4	3	3	4	2	3	2	1

		1	2	3	4	5	6	7	8	9
8.c:	A:	4	3	3	4	2	3	2	1	

		1	2	3	4	5	6	7	8	9
8.d:	A:	4	4	3	3	1	2	3	2	

		1	2	3	4	5	6	7	8	9
8.e:	A:	4	4	3	3	2	3	2	1	

Answers:

- ✔ 9.a: $h'(99) = 0$
- ✔ 9.b: $h'(99) = 1$
- 9.c: $h'(99) = 2$
- 9.d: $h'(99) = 3$
- 9.e: $h'(99) = 4$
- 9.f: $h'(99) = 5$
- ✔ 9.g: $h'(99) = 6$

$\boxed{\text{3 point}}$

Vi ser på en hashtabel H, der bruger linear probing og en auxiliary hashfunktion $h'(x)$. Hashtabellen ser lige nu sådan ud:

	0	1	2	3	4	5	6
H:	33		27	32	55		47

Derefter indsættes 99, hvorefter hashtabellen ser sådan ud:

	0	1	2	3	4	5	6
H:	33	99	27	32	55		47

Hvilke af følgende værdier af $h'(99)$ er mulige? [*Et eller flere svar.*]

4 point

Vi ser på en hashtabel H, der bruger double hashing og en auxiliary hash-funktioner $h_1(x)$ og $h_2(x)$. Hashtabellen ser lige nu sådan ud:

	0	1	2	3	4	5	6
H:	33		27	32	55		47

Derefter indsættes 99, hvorefter hashtabellen ser sådan ud:

	0	1	2	3	4	5	6
H:	33	99	27	32	55		47

Hvis $h_1(99) = 2$, hvilke af følgende værdier af $h_2(99)$ er da mulige? [*Et eller flere svar.*]

Selected Answers:

10.e: $h_2(99) = 5$

Answers:

10.a: $h_2(99) = 1$

10.b: $h_2(99) = 2$

10.c: $h_2(99) = 3$

10.d: $h_2(99) = 4$

10.e: $h_2(99) = 5$

10.f: $h_2(99) = 6$

6 point

Udfør PARTITION(A,1,7) på nedenstående array.

	1	2	3	4	5	6	7
A:	6	2	4	5	1	7	3

Hvilket af nedenstående svar angiver udseendet af A bagefter?

Selected Answer:

11.a:

	1	2	3	4	5	6	7
A:	2	1	3	5	6	7	4

Answers:

11.a:

	1	2	3	4	5	6	7
A:	2	1	3	5	6	7	4

11.b:

	1	2	3	4	5	6	7
A:	2	1	3	6	4	5	7

11.c:

	1	2	3	4	5	6	7
A:	1	2	3	4	5	6	7

11.d:

	1	2	3	4	5	6	7
A:	3	2	1	4	5	6	7

11.e:

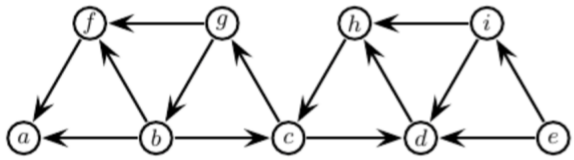
	1	2	3	4	5	6	7
A:	1	2	3	7	6	5	4

11.f:

	1	2	3	4	5	6	7
A:	2	1	4	5	6	7	3

3 point

Hvor mange stærke sammenhængskomponenter har grafen G_1 nedenfor?



G_1

Selected Answer:

12.d: 5

Answers:

12.a: 2

12.b: 3

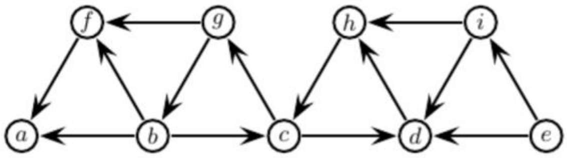
12.c: 4

12.d: 5

12.e: 6

$\boxed{\text{3 point}}$

Vi ser stadig på grafen G_1.



G_1

For hvilken af følgende orienterede kanter gælder det, at antal stærke sammenhængskomponenter falder til én hvis kanten tilføjes til G_1? [En kant fra u til v skrives som sædvanligt (u, v).]

Selected Answer:

13.e: (a, e)

Answers:

13.a: (h, g)

13.b: (g, h)

13.c: (f, i)

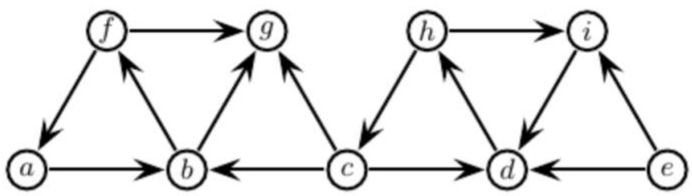
13.d: (i, f)

13.e: (a, e)

13.f: (e, a)

3 point

Udfør bredde-først søgning BFS(G_2, c) på grafen G_2 nedenfor med start i knuden c.
For BFS afhænger udførelsen af ordningen af knuders nabolister. Du skal i dette eksamenssæt antage, at en knudes naboliste er sorteret i alfabetisk orden efter naboknudernes navne.



G_2

Startknuden c tildeles som altid d-værdien 0. Hvilken knude er den *første*, som under udførelsen tildeles d-værdien 2?

Selected Answer: 14.e: Knuden f

Answers:

14.a: Knuden a

14.b: Knuden b

14.c: Knuden d

14.d: Knuden e

14.e: Knuden f

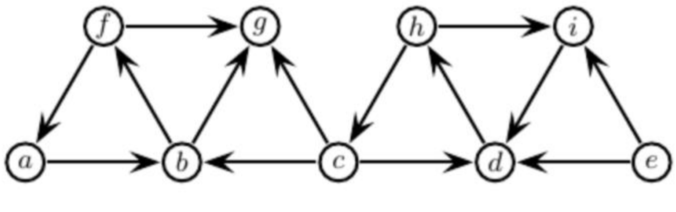
14.f: Knuden g

14.g: Knuden h

14.h: Knuden i

$\boxed{\text{3 point}}$

Vi ser stadig på bredde-først søgning på grafen G_2 nedenfor med start i knuden c.



G_2

Hvilken knude er den *sidste*, som under udførelsen tildeles d-værdien 3?

Answers:

15.a: Knuden a

15.b: Knuden b

15.c: Knuden d

15.d: Knuden e

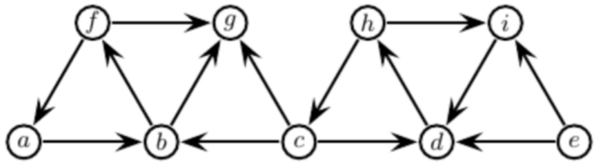
15.e: Knuden f

15.f: Knuden g

15.g: Knuden h

15.h: Knuden i

Vi ser stadig på bredde-først søgning på grafen G_2 nedenfor med start i knuden c.



G_2

Hvor mange knuder har ved algoritmens afslutning d-værdien ∞?

Selected Answer: 16.b: 1

Answers:

16.a: 0

16.b: 1

16.c: 2

16.d: 3

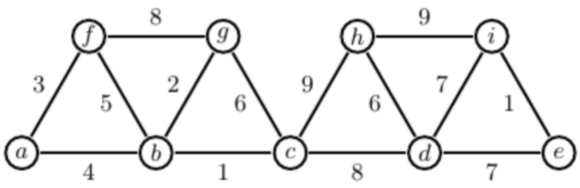
16.e: 4

16.f: 5

16.g: 6

$\boxed{\text{4 point}}$

Udfør Kruskals algoritme på grafen G_3 nedenfor.



G_3

Hvilken kant er den første, som undersøges af algoritmen, men ikke tages med i MST?

Selected Answer:

17.a: Kanten (b, f)

Answers:

17.a: Kanten (b, f)

17.b: Kanten (c, g)

17.c: Kanten (d, i)

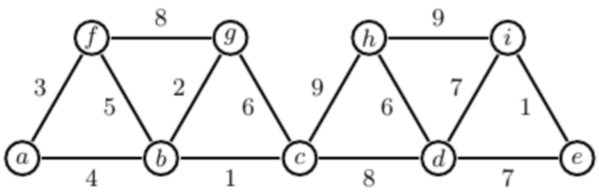
17.d: Kanten (d, e)

17.e: Kanten (f, g)

17.f: Kanten (h, i)

$\boxed{\text{3 point}}$

Vi ser stadig på Kruskals algoritme på grafen G_3.



G_3

Hvor mange sammenhængskomponenter har grafen (V, A) efter at otte kanter er blevet undersøgt af algoritmen?
[Her er V knuderne i G_3, og A er de kanter, som indtil nu er taget med i MST.]

Selected Answer: [None Given]

Answers:

18.a: 1

18.b: 2

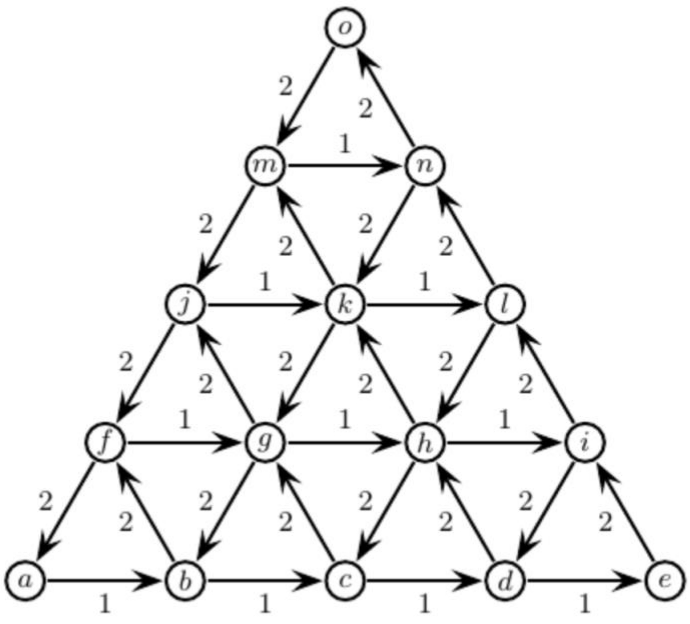
18.c: 3

18.d: 4

18.e: 5

18.f: 6

Hvilke af nedenstående algoritmer kan bruges til at finde korteste veje på grafen G_4? [*Et eller flere svar.*]



G_4

Selected Answers:

- ✓ 19.a: DIJKSTRA
- ✓ 19.b: BELLMAN-FORD
- ✗ 19.c: DAG-SHORTEST-PATHS

Answers:

- ✓ 19.a: DIJKSTRA
- ✓ 19.b: BELLMAN-FORD
- 19.c: DAG-SHORTEST-PATHS
- 19.d: PRIM
- 19.e: BREADTH-FIRST-SEARCH
- 19.f: DEPTH-FIRST-SEARCH

På grafer med n knuder og $m = 3n$ kanter ønsker vi at finde de korteste veje mellem alle par af knuder. Dette kan både løses ved at køre algoritmen BELLMAN-FORD n gange (med start i hver af de n knuder) og ved at køre algoritmen FLOYD-WARSHALL én gang.
Hvilken af disse to metoder har på denne type grafer den bedste asymptotiske køretid som funktion af n?

Selected Answer:

20.b: FLOYD-WARSHALL

Answers:

20.a: BELLMAN-FORD kørt n gange

20.b: FLOYD-WARSHALL

20.c: De har samme asymptotiske køretid.

4 point

En fil indeholder nedenstående tegn med de angivne hyppigheder.

Tegn	a	b	c	d	e	f
Hyppighed	500	400	300	250	200	150

Lav et Huffman-træ på dette input. Hvor mange bits er der i kodeordet for `c`?

Selected Answer:

21.c: 3

Answers:

21.a: 1

21.b: 2

21.c: 3

21.d: 4

21.e: 5

3 point

Vi ser stadig på et Huffman-træ på nedenstående input.

Tegn	a	b	c	d	e	f
Hyppighed	500	400	300	250	200	150

Hvor mange bits fylder strengen `caffebad`, hvis den kodes med dette Huffman-træ?

Selected Answer:

22.d: 21

Answers:

22.a: 18

22.b: 19

22.c: 20

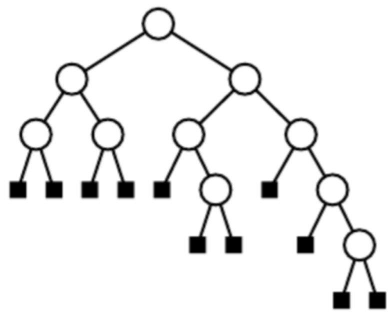
22.d: 21

22.e: 22

22.f: 23

22.g: 24

På hvor mange måder kan man i nedenstående træ farve knuderne, så det bliver et lovligt rød-sort træ?



Selected Answer: [None Given]

Answers:

23.a: På ingen måder.

23.b: På én måde.

23.c: På to måder.

23.d: På tre måder.

23.e: På fire måder.

$\boxed{\text{2 point}}$

For følgende algoritme, hvad er den asymptotiske køretid i Θ-notation som funktion af n?

```
ALGORITME1(n)
    i = 1
    j = n
    while i < j
        i = i + 1
        j = j - 1
```

Selected Answer:

24.c: $\Theta(n)$

Answers:

24.a: $\Theta(\log n)$

24.b: $\Theta((\log n)^2)$

24.c: $\Theta(n)$

24.d: $\Theta(n \log n)$

24.e: $\Theta(n(\log n)^2)$

24.f: $\Theta(n^2)$

24.g: $\Theta(2^n)$

$\boxed{\text{2 point}}$

For følgende algoritme, hvad er den asymptotiske køretid i Θ-notation som funktion af n?

$$
\begin{array}{l}
\textsc{Algoritme2}(n) \\
\quad \textbf{for } i = 1 \textbf{ to } n \\
\quad\quad j = n - i \\
\quad\quad \textbf{while } j < n \\
\quad\quad\quad j = j + 1
\end{array}
$$

Selected Answer:

25.f: $\Theta(n^2)$

Answers:

25.a: $\Theta(\log n)$

25.b: $\Theta((\log n)^2)$

25.c: $\Theta(n)$

25.d: $\Theta(n \log n)$

25.e: $\Theta(n(\log n)^2)$

25.f: $\Theta(n^2)$

25.g: $\Theta(2^n)$

2 point

For følgende algoritme, hvad er den asymptotiske køretid i Θ-notation som funktion af n?

$$
\begin{array}{l}
\textsc{Algoritme3}(n) \\
\quad i = n \\
\quad \textbf{while } i > 1 \\
\quad\quad j = 1 \\
\quad\quad \textbf{while } j < i \\
\quad\quad\quad j = 2 \cdot j \\
\quad\quad i = i - 1
\end{array}
$$

Selected Answer:

26.d: $\Theta(n \log n)$

Answers:

26.a: $\Theta(\log n)$

26.b: $\Theta((\log n)^2)$

26.c: $\Theta(n)$

26.d: $\Theta(n \log n)$

26.e: $\Theta(n(\log n)^2)$

26.f: $\Theta(n^2)$

26.g: $\Theta(2^n)$

2 point

For følgende algoritme, hvad er den asymptotiske køretid i Θ-notation som funktion af n?

$$
\begin{array}{l}
\textsc{Algoritme4}(n) \\
\quad i = 1 \\
\quad \textbf{while } i < n \\
\quad\quad j = i \\
\quad\quad \textbf{while } j > 1 \\
\quad\quad\quad j = j/2 \\
\quad\quad i = 2 \cdot i
\end{array}
$$

Selected Answer:

27.a: $\Theta(\log n)$

Answers:

27.a: $\Theta(\log n)$

27.b: $\Theta((\log n)^2)$

27.c: $\Theta(n)$

27.d: $\Theta(n \log n)$

27.e: $\Theta(n(\log n)^2)$

27.f: $\Theta(n^2)$

27.g: $\Theta(2^n)$

5 point

Vi ser i denne opgaver på at sortere n heltal, som antager værdier i intervallet $[0, n^2[$.
Med TREESORT mener vi den algoritme, som indsætter tallene et efter et i et søgetræ og derefter laver et inorder gennemløb.
Hvilke af nedenstående algoritmer har worst case køretid $\Theta(n^2)$? [*Et eller flere svar.*]

Selected Answers:

28.e: TREESORT, hvor det binære søgetræ er et ubalanceret søgetræ.

28.f: QUICKSORT

28.g: INSERTIONSORT

Answers:

28.a: COUNTINGSORT

28.b: RADIXSORT, hvor heltal betragtes som bestående af to digits med værdier i intervallet $[0, n[$.

28.c: MERGESORT

28.d: TREESORT, hvor det binære søgetræ er et rød-sort træ.

28.e: TREESORT, hvor det binære søgetræ er et ubalanceret søgetræ.

28.f: QUICKSORT

28.g: INSERTIONSORT

$\boxed{\text{5 point}}$

Hvad er worst case køretiden for HEAPSORT, når den køres på n identiske elementer?

Selected Answer:

29.c: $O(n)$

Answers:

29.a: $O(1)$

29.b: $O(\log n)$

29.c: $O(n)$

29.d: $O(n \log n)$

29.e: $O(n^2)$

$\boxed{\text{3 point}}$

For et bestemt optimeringsproblem (som her ikke beskrives nærmere) er input givet ved værdier c_k for $k = 1, 2, \ldots, n$.
Det oplyses, at en løsning $L(i, j)$ til problemet kan beskrives med denne rekursionsligning:

$$L(i,j) = \begin{cases} c_i & \text{hvis } i = j \\ i \cdot L(i+1, j) + j \cdot L(i, j-1) & \text{hvis } i < j \end{cases}$$

Hvis $L(1, n)$ findes via dynamisk programmering baseret på ovenstående rekursionsligning, hvilken af nedenstående køretider opnås?

Selected Answer: [None Given]

Answers:

30.a: $\Theta(1)$

30.b: $\Theta(n^{1/2})$

30.c: $\Theta(n)$

30.d: $\Theta(n^{3/2})$

30.e: $\Theta(n^2)$

30.f: $\Theta(n^3)$

$\boxed{\text{2 point}}$

Vi ser stadig på samme problem, hvis løsning $L(i, j)$ kan beskrives med denne rekursionsligning:

$$L(i,j) = \begin{cases} c_i & \text{hvis } i = j \\ i \cdot L(i+1, j) + j \cdot L(i, j-1) & \text{hvis } i < j \end{cases}$$

Hvis $L(1, n)$ findes via dynamisk programmering baseret på ovenstående rekursionsligning, hvad er det mindste pladsforbrug, som kan opnås?

Selected Answer: [None Given]

Answers:

31.a: $\Theta(1)$

31.b: $\Theta(n^{1/2})$

31.c: $\Theta(n)$

31.d: $\Theta(n^{3/2})$

31.e: $\Theta(n^2)$

31.f: $\Theta(n^3)$