Hvilket af nedenstående svar gælder for følgende rekursionsligning?

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

Selected Answer: [None Given]

Answers:

1.a: $T(n) = \Theta(\log n)$.

1.b: $T(n) = \Theta(n^{1/4})$.

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

1.d: $T(n) = \Theta(n \log n)$.

1.e: $T(n) = \Theta(n^4)$.

1.f: Rekursionsligningen kan ikke løses med Master Theorem.

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

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

Selected Answer: [None Given]

Answers:

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

2.b: $T(n) = \Theta(n^{3/4})$.

2.c: $T(n) = \Theta(n^{\alpha})$ med $\alpha = \log_4(3)$.

2.d: $T(n) = \Theta(n)$.

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

2.f: $T(n) = \Theta(n^{4/3})$.

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

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = 3T(n/4) + n^{3/4}$$

Selected Answer: [None Given]

Answers:

3.a: $T(n) = \Theta(\log n)$.

3.b: $T(n) = \Theta(n^{3/4})$.

3.c: $T(n) = \Theta(n^{\alpha})$ med $\alpha = \log_4(3)$.

3.d: $T(n) = \Theta(n)$.

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

3.f: $T(n) = \Theta(n^{4/3})$.

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

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = 3T(n^{1/4}) + n^{3/4}$$

Selected Answer: [None Given]

Answers:

4.a: $T(n) = \Theta(\log n)$.

4.b: $T(n) = \Theta(n^{\alpha})$ med $\alpha = \log_4(3)$.

4.c: $T(n) = \Theta(n)$.

4.d: $T(n) = \Theta(n \log n)$.

4.e: $T(n) = \Theta(n^{4/3})$.

4.f: Rekursionsligningen kan ikke løses med Master Theorem.

5.a: n^2 er $O(n^3)$

5.b: n^2 er $o(n^3)$

5.c: n^2 er $\Theta(3n^2 + 2n^3)$

5.d: 2^n er $O(n^4)$

5.e: n^2 er $O(4^n)$

5.f: $(\log n)^4$ er $O(n/(\log n)^4)$

5.g: 4^n er $\omega(2^n)$

5.h: $(1/2)^n$ er $O((1/4)^n)$

5.i: $1/\log n$ er $O(1)$

5.j: $\sin n$ er $O(\log n)$

Hvilke af nedenstående arrays er en min-heap? [*Et eller flere svar.*]

Selected Answers: [None Given]

Answers:

6.a: A:

1	2	3	4	5	6	7	8
0	1	0	1	1	0	1	1

6.b: A:

1	2	3	4	5	6	7	8
0	0	0	1	1	1	0	1

6.c: A:

1	2	3	4	5	6	7	8
0	0	1	0	0	1	1	0

6.d: A:

1	2	3	4	5	6	7	8
0	1	0	1	0	1	0	1

6.e: A:

1	2	3	4	5	6	7	8
0	0	0	0	1	1	1	1

6.f: A:

1	2	3	4	5	6	7	8
1	1	1	1	0	0	0	0

6.g: A:

1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0

Udfør to HEAP-EXTRACT-MAX(A) på nedenstående max-heap A.

	1	2	3	4	5	6	7	8
A:	1	1	0	1	0	0	0	1

Hvilket af nedenstående svar angiver udseendet af heapen efter disse to operationer?

Selected Answer: [None Given]

Answers:

		1	2	3	4	5	6	7	8
7.a:	A:	0	1	0	1	0	0		

		1	2	3	4	5	6	7	8
7.b:	A:	1	0	1	0	0	0		

		1	2	3	4	5	6	7	8
7.c:	A:	1	1	0	0	0	0		

		1	2	3	4	5	6	7	8
7.d:	A:	1	0	0	1	0	0		

		1	2	3	4	5	6	7	8
7.e:	A:	1	1	0	1	0	0	0	

For en funktion $h'(x)$ gælder

$$\begin{array}{lcl} h'(7) & = & 2 \\ h'(9) & = & 2 \\ h'(13) & = & 3 \\ h'(17) & = & 4 \end{array}$$

Vi ser på en hashtabel H, der bruger linear probing med ovennævnte funktion $h'(x)$ som auxiliary hashfunktion. Startende med en tom hashtabel er tallene $\{7, 9, 13, 17\}$ blevet indsat i en eller anden rækkefølge. Resultatet er blevet:

	0	1	2	3	4	5	6	7	8	9	10
H:			9	7	17	13					

Hvilke af nedenstående rækkefølger for indsættelse kan der være tale om? [*Et eller flere svar.*]

8.a: $7, 9, 13, 17$

8.b: $13, 7, 17, 9$

✓ 8.c: $9, 17, 7, 13$

8.d: $17, 13, 9, 7$

✓ 8.e: $9, 7, 17, 13$

Vi ønsker at bruge RADIX-SORT(A,4) til at sortere nedenstående array i stigende orden.

	1	2	3	4	5	6	7
A:	2452	5363	4433	1413	2433	3222	2121

Hvad er indholdet af indholdet af A efter udførelsen af *tre* af de fire iterationer i RADIX-SORT(A,4)?

Selected Answer: [None Given]

Answers:

		1	2	3	4	5	6	7
9.a:	A:	2452	5363	4433	1413	2433	3222	2121

		1	2	3	4	5	6	7
9.b:	A:	2121	3222	5363	4433	2452	1413	2433

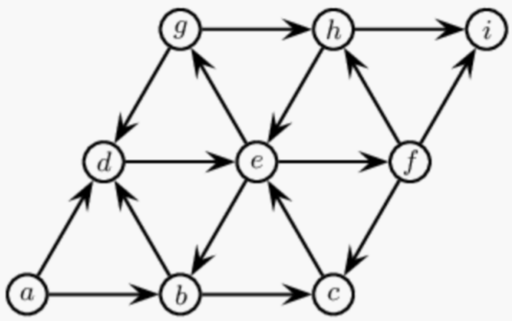
		1	2	3	4	5	6	7
9.c:	A:	1413	2121	3222	4433	2433	2452	5363

		1	2	3	4	5	6	7
9.d:	A:	2121	3222	5363	1413	4433	2433	2452

		1	2	3	4	5	6	7
9.e:	A:	1413	2121	2433	2452	3222	4433	5363

		1	2	3	4	5	6	7
9.f:	A:	2121	2452	3222	5363	4433	1413	2433

I grafen G_1 nedenfor, for hvilke af følgende par af knuder ligger begge knuder i samme stærke sammenhængskomponent? [*Et eller flere svar.*]



G_1

Selected Answers: [None Given]

Answers:

10.a: a og f

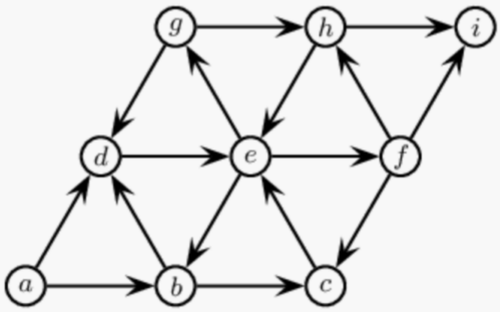
10.b: b og h

10.c: c og d

10.d: b og i

10.e: a og i

Vi ser stadig på grafen G_1. Hvor mange stærke sammenhængskomponenter er der i grafen?



G_1

Selected Answer: [None Given]

Answers:

11.a: 1

11.b: 2

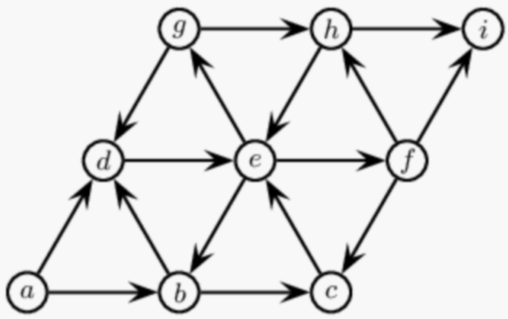
11.c: 3

11.d: 4

11.e: 5

11.f: 6

Vi ser stadig på grafen G_1. Udfør dybde-først søgning (DFS) med start i knuden a.
For DFS afhænger resultatet af ordningen af knuders nabolister. Du skal her antage at en knudes naboliste er sorteret i alfabetisk orden efter naboknudernes navne.
Hvilken knude får starttiden (discovery time) 12?



G_1

Selected Answer: [None Given]

Answers:

12.a: Knuden d

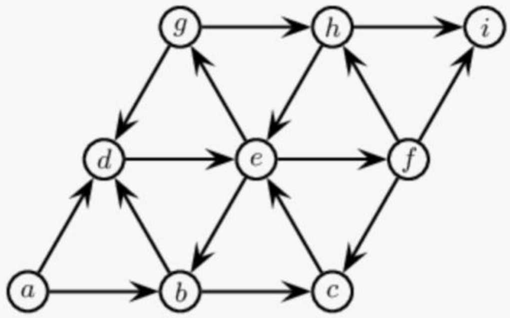
12.b: Knuden e

12.c: Knuden f

12.d: Knuden g

12.e: Ingen knude får starttid 12.

Vi ser stadig på dybde-først søgning (DFS) på grafen G_1 med start i knuden a. Du skal stadig antage at en knudes naboliste er sorteret i alfabetisk orden efter naboknudernes navne.
Hvilken knude får sluttiden (finishing time) 10?



G_1

Selected Answer: [None Given]

Answers:

13.a: Knuden b

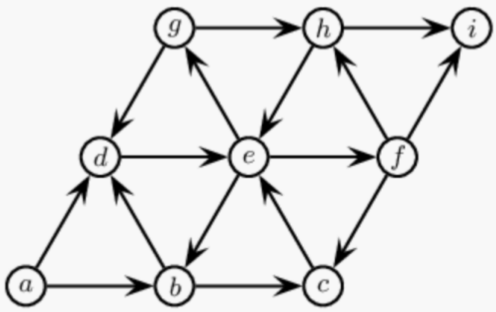
13.b: Knuden d

13.c: Knuden f

13.d: Knuden h

13.e: Ingen knude får sluttid 10.

Vi ser stadig på dybde-først søgning (DFS) på grafen G_1 med start i knuden a. Du skal stadig antage at en knudes naboliste er sorteret i alfabetisk orden efter naboknudernes navne.
Bestem for alle kanter deres type (tree edge, back edge, forward edge, cross edge).
Hvilket af svarene nedenfor indeholder præcis én kant af hver type?



G_1

Selected Answer: [None Given]

Answers:

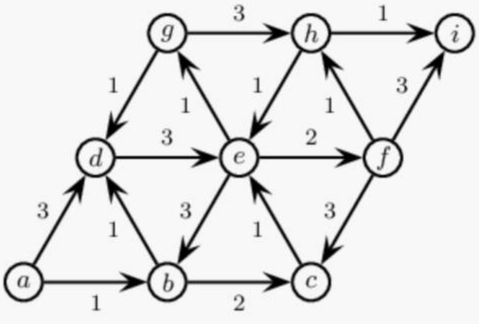
14.a: (c, e), (e, g), (h, e), (e, b)

14.b: (g, h), (h, i), (d, e), (e, f)

14.c: (b, d), (d, e), (g, d), (g, h)

14.d: (a, b), (f, c), (f, i), (f, h)

Udfør Dijkstras algoritme på grafen G_2 nedenfor, med start i knuden a.
Den første knude, som udtages fra prioritetskøen (med operationen EXTRACT-MIN) under algoritmens kørsel, er knuden a. Hvilken knude er den sjette som udtages?
(Hvis der på et tidspunkt er flere knuder i prioritetskøen med den laveste nøgleværdi, udtages den knude blandt dem, som har det alfabetisk set mindste navn.)



G_2

Selected Answer: [None Given]

Answers:

15.a: Knuden d

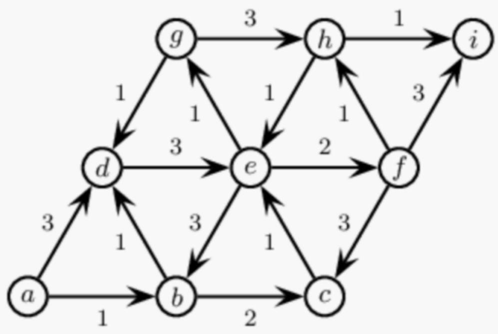
15.b: Knuden e

15.c: Knuden f

15.d: Knuden g

15.e: Knuden h

Vi ser stadig på Dijkstras algoritme udførst på grafen G_2 med start i knuden a.
Hvad er længden af en korteste vej fra knuden a til knuden i?



G_2

Selected Answer: [None Given]

Answers:

16.a: 6

16.b: 7

16.c: 8

16.d: 9

16.e: 10

På grafer med n knuder og $m = n(n-1)$ kanter, ønsker vi at finde de korteste veje mellem alle par af knuder. Dette kan både løses ved at køre algoritmen Dijkstra n gange (med start i hver af de n knuder), og ved at køre algoritmen Floyd-Warshall én gang. Vi antager, at i algoritmen Dijkstra er prioritetskøen implementeret med en binær heap.
Hvilken af disse to metoder har på denne type grafer den bedste asymptotiske køretid som funktion af n?

Selected Answer: [None Given]

Answers:

17.a: Dijkstra kørt n gange

17.b: Floyd-Warshall

17.c: De har samme asymptotiske køretid.

En fil indeholder nedenstående tegn med de angivne hyppigheder. Der er 2800 tegn i alt.

Tegn	a	b	c	d	e	f	g
Hyppighed	100	200	300	400	500	600	700

Lav et Huffman-træ på dette input. Hvor mange bits fylder den kodede fil (dvs. hvad er den samlede længde i bits af de 2800 kodede tegn)?

Selected Answer: [None Given]

Answers:

18.a: 7300

18.b: 7400

18.c: 7500

18.d: 7600

18.e: 8100

18.f: 8300

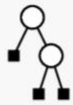
18.g: 8500

For hvilke af de nedenstående træer kan knuderne farves, så træet bliver til et lovligt rød-sort træ? [*Et eller flere svar.*]

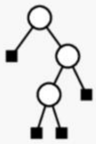
Selected Answers: [None Given]

Answers:

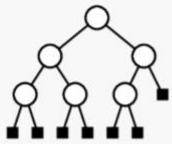
19.a:



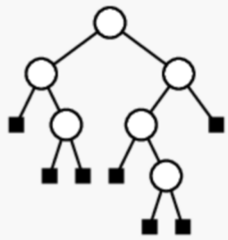
19.b:



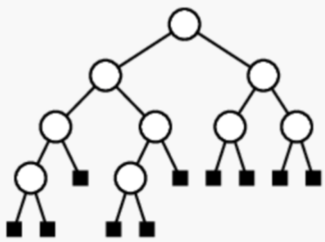
19.c:



19.d:



19.e:



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

$\textsc{Algoritme1}(n)$
    $s = 0$
    for $i = 1$ **to** n
        for $j = 1$ **to** n
            if $i == j$
                for $k = 1$ **to** n
                    $s = s + 1$

Selected Answer: [None Given]

Answers:

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

20.b: $\Theta(n)$

20.c: $\Theta(n \log n)$

20.d: $\Theta(n^2)$

20.e: $\Theta(n^3)$

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

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

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

Selected Answer: [None Given]

Answers:

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

21.b: $\Theta(n)$

21.c: $\Theta(n \log n)$

21.d: $\Theta(n^2)$

21.e: $\Theta(n^3)$

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

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

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

Selected Answer: [None Given]

Answers:

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

22.b: $\Theta(n)$

22.c: $\Theta(n \log n)$

22.d: $\Theta(n^2)$

22.e: $\Theta(n^3)$

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

$\boxed{\text{6 point}}$

Følgende kode har til formål at beregne $\lceil \log_2 n \rceil$.

```
LogBaseTo(n)
    x = 1
    r = 0
    while x < n
        x = 2x
        r = r + 1
    return r
```

Hvilke af nedenstående udsagn er en løkke-invariant for algoritmen (dvs. er altid sandt, når testen i starten af **while**-løkken udføres) for alle input, der er heltal $n \geq 1$. [*Et eller flere svar.*]

Selected Answers: [None Given]

Answers:

23.a: $x = r + 1$

23.b: $2^r \cdot \log_2 n = \log_2(n/x)$

23.c: $x \leq n$

23.d: $2^r = x$

23.e: $x < 2n$

For to sekvenser $X = x_1, x_2, \ldots, x_m$ og $Y = y_1, y_2, \ldots, y_n$ af længde m og n vil vi gerne finde en anden sekvens S, der indeholder både X og Y som delsekvenser. Sådan en sekvens S kalder vi en *supersekvens* for X og Y.
Et eksempel: for X og Y givet ved:

$$X = \mathtt{c\,c\,a\,c\,c}$$
$$Y = \mathtt{a\,c\,a\,c\,a\,a}$$

er følgende S en supersekvens:

$$S = \underline{\mathtt{a}}\,\overline{\underline{\mathtt{c}}}\,\underline{\mathtt{a}}\,\overline{\underline{\mathtt{c}}}\,\overline{\underline{\mathtt{a}}}\,\overline{\mathtt{c}}\,\underline{\mathtt{a}}\,\overline{\mathtt{c}}$$

Delsekvenserne X og Y i S er vist ved henholdsvis over- og understregninger.
For alle par X og Y findes der altid en supersekvens S, for eksempel $S = XY$ (dvs. tegnene i X efterfulgt af tegnene i Y). Dermed findes også altid en *korteste* supersekvens (ikke nødvendigvis entydig, der kan være flere af samme korteste længde).
Givet strengene X og Y lader vi $l(i, j)$ betegne længden af en korteste supersekvens for $X_i = x_1, x_2, \ldots, x_i$ og $Y_j = y_1, y_2, \ldots, y_j$, for $0 \leq i \leq m$ og $0 \leq j \leq n$.
Vi ønsker at bestemme en rekursionsformel for $l(i, j)$. For basistilfældet $i = 0$ (X_i den tomme streng) gælder klart $l(0, j) = j$. For basistilfældet $j = 0$ (Y_j den tomme streng) gælder klart $l(i, 0) = i$.
Hvilke af følgende er korrekte rekursionsformler for det generelle tilfælde med $i > 0$ og $j > 0$? [*Et eller flere svar.*]

24.a:

$$l(i,j) = \begin{cases} 1 + \min\{l(i-1,j), l(i,j-1)\} & \text{hvis } x_i \neq y_j \\ l(i-1,j-1) & \text{hvis } x_i = y_j \end{cases}$$

24.b:

$$l(i,j) = \begin{cases} \min\{l(i-1,j), l(i,j-1)\} & \text{hvis } x_i \neq y_j \\ 1 + l(i-1,j-1) & \text{hvis } x_i = y_j \end{cases}$$

24.c:

$$l(i,j) = \begin{cases} 1 + \min\{l(i-1,j), l(i,j-1)\} & \text{hvis } x_i \neq y_j \\ 1 + l(i-1,j-1) & \text{hvis } x_i = y_j \end{cases}$$

24.d:

$$l(i,j) = \begin{cases} 1 + \min\{l(i-1,j), l(i,j-1)\} & \text{hvis } x_i \neq y_j \\ 2 + l(i-1,j-1) & \text{hvis } x_i = y_j \end{cases}$$

24.e:

$$l(i,j) = \begin{cases} 1 + \min\{l(i-1,j), l(i,j-1)\} & \text{hvis } x_i \neq y_j \\ 1 + \min\{l(i-1,j), l(i,j-1), l(i-1,j-1)\} & \text{hvis } x_i = y_j \end{cases}$$

24.f:

$$l(i,j) = 1 + \min\{l(i-1,j), l(i,j-1), l(i-1,j-1)\}$$

For et andet optimeringsproblem (som her ikke beskrives nærmere) er input givet ved en omkostning c_{ij} (dvs. et tal) for alle i og j. Det oplyses at en løsning $b(i, j)$ til problemet kan beskrives med denne rekursionsligning:

$$b(i,j) = \begin{cases} 0 & \text{hvis } i = 0 \text{ eller } j = 0 \\ c_{ij} + \min\{b(i-1,k) \mid 0 \leq k < j\} & \text{hvis } i > 0 \text{ og } j > 0 \end{cases}$$

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

25.a: $\Theta(1)$

25.b: $\Theta(m)$

25.c: $\Theta(n)$

25.d: $\Theta(mn)$

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

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

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

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

$$b(i,j) = \begin{cases} 0 & \text{hvis } i = 0 \text{ eller } j = 0 \\ c_{ij} + \min\{b(i-1,k) \mid 0 \leq k < j\} & \text{hvis } i > 0 \text{ og } j > 0 \end{cases}$$

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

Selected Answer: [None Given]

Answers:

26.a: $\Theta(1)$

26.b: $\Theta(m)$

26.c: $\Theta(n)$

26.d: $\Theta(mn)$

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

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

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