Spørgsmål 1

2 point

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = 2 \cdot T(n/2) + n^{1/2}$$

Svar 1.1: $T(n) = \Theta(\log n)$

Svar 1.2: $T(n) = \Theta(n^{1/2})$

Svar 1.3: $T(n) = \Theta(n)$

Svar 1.4: $T(n) = \Theta(n \log n)$

Svar 1.5: $T(n) = \Theta(n^{3/2})$

Svar 1.6: $T(n) = \Theta(n^2)$

Svar 1.7: Rekursionsligningen kan ikke løses med Master Theorem.

Spørgsmål 2

2 point

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = T(n/2) + n^{1/2}$$

Svar 2.1: $T(n) = \Theta(\log n)$

Svar 2.2: $T(n) = \Theta(n^{1/2})$

Svar 2.3: $T(n) = \Theta(n)$

Svar 2.4: $T(n) = \Theta(n \log n)$

Svar 2.5: $T(n) = \Theta(n^{3/2})$

Svar 2.6: $T(n) = \Theta(n^2)$

Svar 2.7: Rekursionsligningen kan ikke løses med Master Theorem.

Spørgsmål 3

2 point

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = 2 \cdot T(n/2) + 1/2$$

Svar 3.1: $T(n) = \Theta(\log n)$

Svar 3.2: $T(n) = \Theta(n^{1/2})$

Svar 3.3: $T(n) = \Theta(n)$

Svar 3.4: $T(n) = \Theta(n \log n)$

Svar 3.5: $T(n) = \Theta(n^{3/2})$

Svar 3.6: $T(n) = \Theta(n^2)$

Svar 3.7: Rekursionsligningen kan ikke løses med Master Theorem.

Spørgsmål 4

2 point

Hvilket af nedenstående svar gælder for følgende rekursionsligning?

$$T(n) = T(n/2) + 1/2$$

Svar 4.1: $T(n) = \Theta(\log n)$

Svar 4.2: $T(n) = \Theta(n^{1/2})$

Svar 4.3: $T(n) = \Theta(n)$

Svar 4.4: $T(n) = \Theta(n \log n)$

Svar 4.5: $T(n) = \Theta(n^{3/2})$

Svar 4.6: $T(n) = \Theta(n^2)$

Svar 4.7: Rekursionsligningen kan ikke løses med Master Theorem.

Spørgsmål 5

5 point

Hvilke af nedenstående udsagn er sande? [*Et eller flere svar.*]

Svar 5.1: n er $O(\sqrt{n})$

Svar 5.2: $n + n$ er $O(n \log n)$

Svar 5.3: $n \log n$ er $O(n^{3/2})$

Svar 5.4: $(\log n)^2$ er $O(n^{1/2})$

Svar 5.5: 3^n er $O((\log n)^3)$

Svar 5.6: $2^n \log n$ er $O(2^n n)$

Svar 5.7: $n^{1/7}$ er $O(\log(n^{17}))$

Spørgsmål 6

5 point

Hvilke af nedenstående udsagn er sande? [*Et eller flere svar.*]

Svar 6.1: n er $\Omega((\log n)^2)$

Svar 6.2: 4^n er $\omega(2^n)$

Svar 6.3: $n + n + n$ er $\Theta(n/3)$

Svar 6.4: $(\log n)^3$ er $\Theta(3 \log n)$

Svar 6.5: $n^2 / \log n$ er $o(n^2 \log n)$

Svar 6.6: $n^{1.5} + n^{2.0}$ er $\Theta(n^{1.75})$

Svar 6.7: 2^n er $o(n^n)$

Spørgsmål 7

3 point

Udfør Extract-Min på nedenstående min-heap A:

	1	2	3	4	5	6	7	8	9	10
A:	3	5	6	10	11	8	7	18	16	15

På hvilken plads i A befinder 15 sig bagefter?

Svar 7.1: 1

Svar 7.2: 2

Svar 7.3: 3

Svar 7.4: 4

Svar 7.5: 5

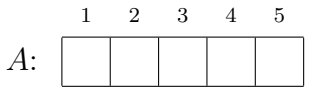
Svar 7.6: 8

Svar 7.7: 9

Spørgsmål 8

3 point

En min-heap A



indeholder følgende fem nøgler:

$$\{1, 2, 3, 4, 5\}$$

Angiv alle pladser i A, hvor det er muligt at nøglen 4 kan stå. [*Et eller flere svar.*]

Svar 8.1: 1

Svar 8.2: 2

Svar 8.3: 3

Svar 8.4: 4

Svar 8.5: 5

Spørgsmål 9

2 point

For en funktion $h_1(x)$ gælder

$$\begin{aligned} h_1(22) &= 6 \\ h_1(33) &= 1 \\ h_1(44) &= 4 \\ h_1(55) &= 1 \\ h_1(66) &= 6 \\ h_1(77) &= 1 \end{aligned}$$

Vi ser på en hashtabel H af længde syv, der bruger linear probing med ovennævnte funktion $h_1(x)$ som auxiliary hashfunktion. Startende med en tom hashtabel er tallene $\{22, 33, 44, 55, 66, 77\}$ blevet indsat i en eller anden rækkefølge. Resultatet er blevet:

	0	1	2	3	4	5	6
H:	22	77	55	33	44		66

Hvilke af tallene kan være blevet indsat først (dvs. kan have stået først i indsættelsesrækkefølgen)? [*Et eller flere svar.*]

Svar 9.1: 22

Svar 9.2: 33

Svar 9.3: 44

Svar 9.4: 55

Svar 9.5: 66

Svar 9.6: 77

Spørgsmål 10

3 point

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

	0	1	2	3	4	5	6	7
H:	13	56		32	91		82	

Derefter indsættes 25, hvorefter hashtabellen ser sådan ud:

	0	1	2	3	4	5	6	7
H:	13	56		32	91	25	82	

Hvis $h_1(25) = 3$, hvilke af følgende værdier af $h_2(25)$ er da mulige? [*Et eller flere svar.*]

Svar 10.1: $h_2(25) = 1$

Svar 10.2: $h_2(25) = 2$

Svar 10.3: $h_2(25) = 3$

Svar 10.4: $h_2(25) = 4$

Svar 10.5: $h_2(25) = 5$

Svar 10.6: $h_2(25) = 6$

Svar 10.7: $h_2(25) = 7$

Spørgsmål 11

6 point

Nedenfor er et array af ni heltal med værdier mellem 0 og 6 (inklusive).

	1	2	3	4	5	6	7	8	9
A:	2	0	6	2	3	5	5	1	2

Vi sorterer nu tallene ved at udføre COUNTING-SORT(A,9,6).

I algoritmen bruges et array C med syv pladser (indekseret fra 0 til 6). Hver plads indeholder en tæller, dvs. et heltal.

Hvad er summen af disse syv heltal i C ved algoritmens afslutning?

Svar 11.1: 0

Svar 11.2: 6

Svar 11.3: 9

Svar 11.4: 22

Svar 11.5: 28

Svar 11.6: 37

Spørgsmål 12

4 point

En fil indeholder nedenstående tegn med de angivne hyppigheder.

Tegn	`b`	`c`	`d`	`f`	`g`	`h`
Hyppighed	90	15	40	30	125	35

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

Svar 12.1: 1

Svar 12.2: 2

Svar 12.3: 3

Svar 12.4: 4

Svar 12.5: 5

Spørgsmål 13

3 point

Vi ser stadig på et Huffman-træ for en fil med nedenstående hyppigheder. Der er $90 + 15 + 40 + 30 + 125 + 35 = 335$ tegn i filen.

Tegn	`b`	`c`	`d`	`f`	`g`	`h`
Hyppighed	90	15	40	30	125	35

Hvor mange bits fylder disse 335 tegn i alt hvis de kodes med Huffman-koder?

Svar 13.1: 770

Svar 13.2: 775

Svar 13.3: 785

Svar 13.4: 790

Svar 13.5: 795

Svar 13.6: 820

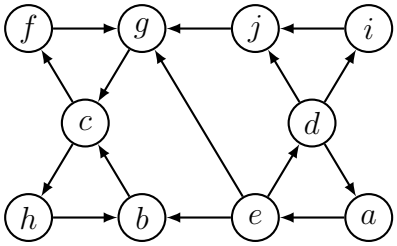
Svar 13.7: 1060

Spørgsmål 14

6 point

Udfør bredde-først søgning BFS(G, a) på grafen nedenfor med start i knuden a.

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.



Hvilken knude er den første, som får tildelt d-værdien 4?

Svar 14.1: c

Svar 14.2: f

Svar 14.3: h

Svar 14.4: i

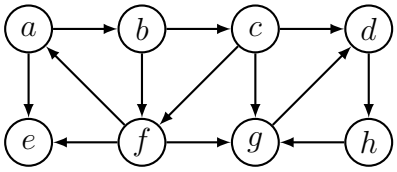
Svar 14.5: j

Spørgsmål 15

5 point

Udfør dybde-først søgning DFS-Visit(G, a) på grafen nedenfor med start i knuden a. Læs også næste spørgsmål, inden at du udfører algoritmen.

For DFS 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.



Hvilken af nedenstående knuder er den sidste, som opdages (dvs. som får den højeste discovery time).

Svar 15.1: d

Svar 15.2: e

Svar 15.3: f

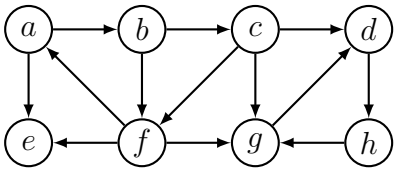
Svar 15.4: g

Svar 15.5: h

Spørgsmål 16

2 point

Vi ser stadig på dybde-først søgning DFS-Visit(G, a) på grafen nedenfor med start i knuden a.



Hvor mange forward edges findes i grafen under denne søgning?

Svar 16.1: 0

Svar 16.2: 1

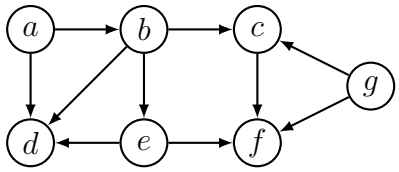
Svar 16.3: 2

Svar 16.4: 3

Svar 16.5: 4

Spørgsmål 17

4 point



Hvilke af nedenstående lister angiver en topologisk sortering af grafen ovenfor? [*Et eller flere svar.*]

Svar 17.1: a, b, c, d, e, f, g

Svar 17.2: a, b, e, d, c, g, f

Svar 17.3: a, b, e, d, g, c, f

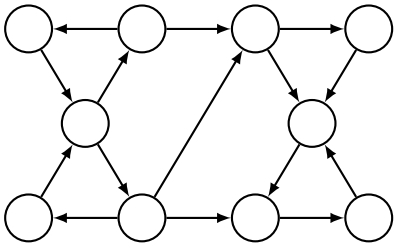
Svar 17.4: a, b, g, c, e, d, f

Svar 17.5: g, a, b, e, c, f, d

Spørgsmål 18

4 point

Hvor mange stærke sammenhængskomponenter har grafen nedenfor?



Svar 18.1: 1

Svar 18.2: 2

Svar 18.3: 3

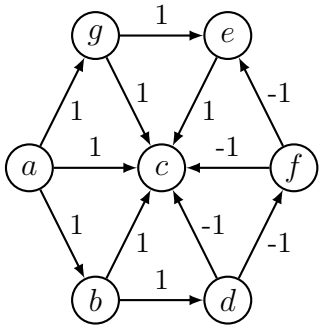
Svar 18.4: 4

Svar 18.5: 5

Spørgsmål 19

4 point

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



Svar 19.1: Dijkstra

Svar 19.2: Bellman-Ford

Svar 19.3: Dag-Shortest-Paths

Svar 19.4: Kruskal

Svar 19.5: Breadth-First-Search

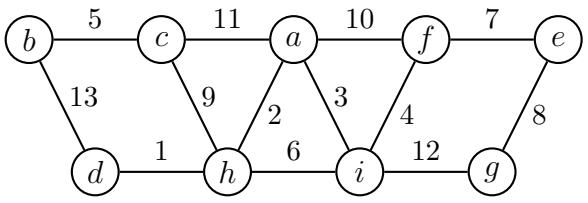
Svar 19.6: Depth-First-Search

Svar 19.7: Floyd-Warshall

Spørgsmål 20

6 point

Brug Prims algoritme til at finde finde et minimum udspændende træ for nedenstående graf. Algoritmen skal starte i knuden a.



Hvilken knude er den sidste, som kommer med i MST?

Svar 20.1: b

Svar 20.2: d

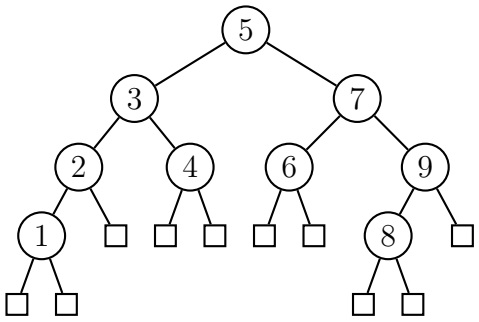
Svar 20.3: e

Svar 20.4: g

Spørgsmål 21

3 point

Hvilke af nedenstående delmængder af knuder vil gøre træet til et lovligt rød-sort træ, hvis netop denne delmænge af knuder farves røde (og resten farves sorte). [*Et eller flere svar.*]



Svar 21.1: $1, 3, 6, 8, 9$

Svar 21.2: $1, 5, 8$

Svar 21.3: $1, 3, 7, 8$

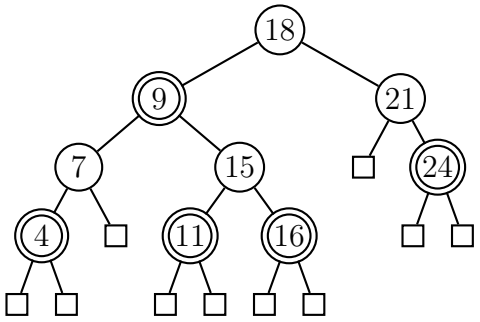
Svar 21.4: $1, 8$

Svar 21.5: $1, 2, 4, 7, 8$

Spørgsmål 22

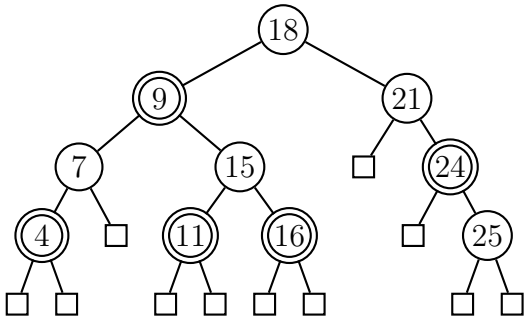
4 point

Indsæt nøglen 25 i nedenstående rød-sorte træ (dobbeltcirkler angiver røde knuder).

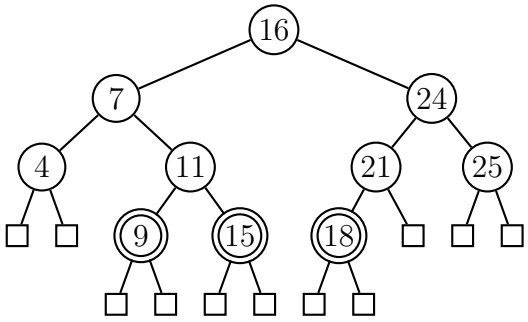


Hvilket er følgende træer viser træet efter indsættelsen?

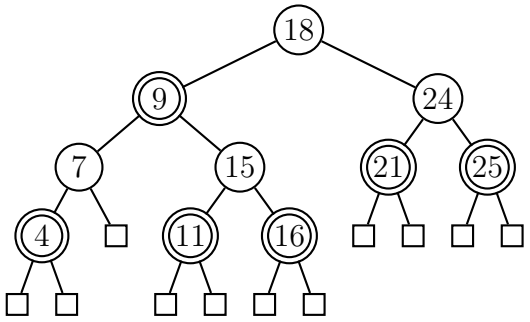
Svar 22.1:



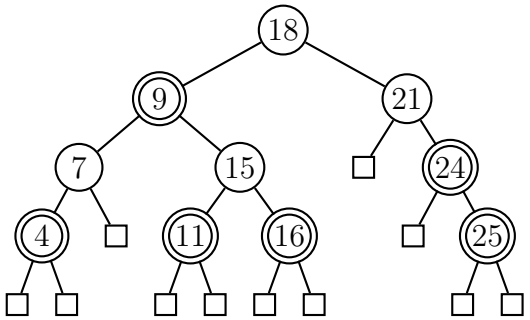
Svar 22.2:



Svar 22.3:



Svar 22.4:



Spørgsmål 23

2 point

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

```
ALGORITME1(n)
    i = 1
    while i ≤ n
        i = i + i
```

Svar 23.1: $\Theta(\log n)$

Svar 23.2: $\Theta(\sqrt{n})$

Svar 23.3: $\Theta(n)$

Svar 23.4: $\Theta(n \log n)$

Svar 23.5: $\Theta(n^2)$

Svar 23.6: $\Theta(n^3)$

Spørgsmål 24

2 point

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

```
ALGORITME2(n)
    i = 1
    j = 1
    while i ≤ n
        i = i + 5
        while j < i
            j = j + 1
```

Svar 24.1: $\Theta(\log n)$

Svar 24.2: $\Theta(\sqrt{n})$

Svar 24.3: $\Theta(n)$

Svar 24.4: $\Theta(n \log n)$

Svar 24.5: $\Theta(n^2)$

Svar 24.6: $\Theta(n^3)$

Spørgsmål 25

2 point

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

```
ALGORITME3(n)
    i = 1
    while i ≤ n
        j = n
        while j > 1
            j = j − 1
        i = 2 ∗ i
```

Svar 25.1: $\Theta(\log n)$

Svar 25.2: $\Theta(\sqrt{n})$

Svar 25.3: $\Theta(n)$

Svar 25.4: $\Theta(n \log n)$

Svar 25.5: $\Theta(n^2)$

Svar 25.6: $\Theta(n^3)$

Spørgsmål 26

2 point

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

```
ALGORITME4(n)
    i = 1
    j = n
    while i ≤ j
        i = 4 * i
        j = 2 * j
```

Svar 26.1: $\Theta(\log n)$

Svar 26.2: $\Theta(\sqrt{n})$

Svar 26.3: $\Theta(n)$

Svar 26.4: $\Theta(n \log n)$

Svar 26.5: $\Theta(n^2)$

Svar 26.6: $\Theta(n^3)$

Spørgsmål 27

2 point

Vi ser i denne opgave på at sortere n heltal med værdier i intervallet $[0\,;n^3[$. Hvad er worst case køretiden for CountingSort på denne type input?

Svar 27.1: $\Theta(n)$

Svar 27.2: $\Theta(n \log n)$

Svar 27.3: $\Theta(n^2)$

Svar 27.4: $\Theta(n^3)$

Spørgsmål 28

2 point

Vi ser stadig på at sortere n heltal med værdier i intervallet $[0\,;n^3[$.

Hvad er worst case køretiden for RadixSort på denne type input, når heltal betragtes som bestående af tre digits med værdier i intervallet $[0\,;n[$?

Svar 28.1: $\Theta(n)$

Svar 28.2: $\Theta(n \log n)$

Svar 28.3: $\Theta(n^2)$

Svar 28.4: $\Theta(n^3)$

Spørgsmål 29

2 point

Vi ser stadig på at sortere n heltal med værdier i intervallet $[0\,;n^3[$.

Hvad er worst case køretiden for QuickSort på denne type input?

Svar 29.1: $\Theta(n)$

Svar 29.2: $\Theta(n \log n)$

Svar 29.3: $\Theta(n^2)$

Svar 29.4: $\Theta(n^3)$

Spørgsmål 30

2 point

Vi har en række tal $x_1 < x_2 < x_3 < \cdots < x_n$, hvor hvert tal enten er hvidt eller sort. En *sort streak* er en sammenhængende delrække $x_i, x_{i+1}, \ldots, x_j$ af sorte tal.

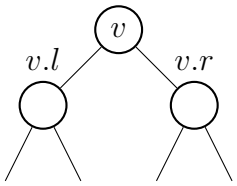
Vi ønsker at kunne indsætte og slette tal i $O(\log n)$ tid og at kunne finde længden af længste sorte streak i $O(1)$ tid.

Vi bruger derfor et rød-sort træ, hvor tallene selv er nøgler, og hvor hver knude v gemmer følgende ekstra information:

v.maxS:	Længden af længste sorte streak i $T(v)$.
v.maxLS:	Længden af længste sorte streak i $T(v)$, som starter ved det mindste tal i $T(v)$.
v.maxRS:	Længden af længste sorte streak i $T(v)$, som ender ved det største tal i $T(v)$.
v.hasWhite:	En boolean, som angiver, om $T(v)$ indeholder et hvidt tal.

Her angiver $T(v)$ rækken af tal, der findes i undertræet med v som rod. Bemærk, at længderne ovenfor godt kan være nul.

Vi ønsker at kunne beregne informationerne i en knude v ud fra informationerne i dens børn $v.l$ og $v.r$.



Vi starter med at se på beregningen af *v.maxS*. Hvilket af følgende svar angiver, hvordan denne kan beregnes ud fra informationen i børnene (vi ser kun på det tilfælde, hvor disse børn begge eksisterer)?

Svar 30.1:

$$v.maxS = \begin{cases} \max\{\, v.l.maxS, v.r.maxS, \\ \qquad v.l.maxRS + v.r.maxLS \,\} & \text{hvis } v\text{'s nøgle er sort} \\ \max\{\, v.l.maxS, v.r.maxS \,\} & \text{hvis } v\text{'s nøgle er hvid} \end{cases}$$

Svar 30.2:

$$v.maxS = \begin{cases} \max\{\, v.l.maxS, v.r.maxS, \\ \qquad v.l.maxRS + 1 + v.r.maxLS \,\} & \text{hvis } v\text{'s nøgle er sort} \\ \max\{\, v.l.maxS, v.r.maxS \,\} & \text{hvis } v\text{'s nøgle er hvid} \end{cases}$$

Svar 30.3:

$$v.maxS = \begin{cases} \max\{\, v.l.maxS, v.r.maxS, \\ \qquad v.l.maxRS + 1 + v.r.maxLS \,\} & \text{hvis } v\text{'s nøgle er hvid} \\ \max\{\, v.l.maxS, v.r.maxS \,\} & \text{hvis } v\text{'s nøgle er sort} \end{cases}$$

Svar 30.4:

$$v.maxS = \begin{cases} v.l.maxS + v.r.maxS \\ \quad + v.l.maxRS + v.r.maxLS & \text{hvis } v\text{'s nøgle er sort} \\ v.l.maxS + v.r.maxS & \text{hvis } v\text{'s nøgle er hvid} \end{cases}$$

Spørgsmål 31

2 point

Vi ser nu på beregningen af *v.maxLS*. [Beregningen af *v.maxRS* foregår helt symmetrisk med *v.maxLS*, og beregningen af *v.hasWhite* er ligetil. Så disse vil vi ikke se på.]

Hvilket af følgende svar angiver, hvordan *v.maxLS* kan beregnes ud fra informationen i børnene (vi ser kun på det tilfælde, hvor disse børn begge eksisterer)?

Svar 31.1:

$$v.maxLS = \begin{cases} v.l.maxLS & \text{hvis } v\text{'s nøgle er hvid} \\ v.l.maxS + 1 + v.r.maxLS & \text{hvis } v\text{'s nøgle er sort} \end{cases}$$

Svar 31.2:

$$v.maxLS = \begin{cases} v.l.maxLS & \text{hvis } v\text{'s nøgle er hvid} \\ v.l.maxS + v.r.maxLS & \text{hvis } v\text{'s nøgle er sort} \end{cases}$$

Svar 31.3:

$$v.maxLS = \begin{cases} v.l.maxLS & \text{hvis } v.l.hasWhite \text{ eller } v\text{'s nøgle er hvid} \\ v.l.maxS + 1 + v.r.maxLS & \text{ellers} \end{cases}$$

Svar 31.4:

$$v.maxLS = \begin{cases} v.l.maxLS & \text{hvis } v.l.hasWhite \text{ eller } v\text{'s nøgle er hvid} \\ v.l.maxS + v.r.maxLS & \text{ellers} \end{cases}$$

Spørgsmål 32

2 point

Via informationen $r.maxS$ i roden r kan vi finde længden af længste sorte streak i $O(1)$ tid.

Vi ønsker nu også at kunne finde en længste sort streak. Mere præcist vil vi gerne i $O(\log n)$ tid kunne finde en knude, hvis nøgle er med i en længste sort streak.

Hvilket af følgende svar angiver en algoritme, som opnår dette? De starter alle med at sætte v lig roden af træet, og vi beskriver kun situationer, hvor begge børn af v eksisterer.

Svar 32.1:

Hvis $v.maxS > v.l.maxS$ og $v.maxS > v.r.maxS$, returner v. Ellers, hvis $v.maxS = v.l.maxS$, fortsæt rekursivt til $v.l$. Ellers fortsæt rekursivt til $v.r$.

Svar 32.2:

Hvis $v.l.maxS > v.r.maxS$, fortsæt rekursivt til $v.l$. Ellers, hvis $v.l.maxS < v.r.maxS$, fortsæt rekursivt til $v.r$. Ellers returner v.

Svar 32.3:

Hvis $v.l.hasWhite$, fortsæt rekursivt til $v.r$. Ellers, hvis $v.r.hasWhite$, fortsæt rekursivt til $v.l$. Ellers returner v.

Svar 32.4:

Hvis ikke $v.hasWhite$, returner v. Ellers, hvis ikke $v.l.hasWhite$, fortsæt rekursivt til $v.l$. Ellers, fortsæt rekursivt til $v.r$.