

Indhold

Asymptotisk notation/ <i>Asymptotic notation</i>	2
Analyse af løkker/ <i>Analysis of loops</i>	7
Indsættelse i søgetræer/ <i>Insertions into search trees</i>	15
Max-Heap-Insert	24
Build-Max-Heap	29
Heap-Extract-Max	34
Partition	39
Radix-sort	44
Lineær probing/ <i>Linear probing</i>	49
Kvadratisk probing/ <i>Quadratic probing</i>	55
Dobbelt hashing/ <i>Double hashing</i>	60
Lovlige rød-sortede træer/ <i>Valid red-black trees</i>	65
Rød-sort træ indsættelse/ <i>Red-black tree insertion</i>	70
Union-find	78
Huffman koder/ <i>Huffman encoding</i>	88
Rekursionsligninger/ <i>Recurrence relations</i>	93
BFS	97
Lovlige BFS træer/ <i>Valid BFS trees</i>	102
DFS	108
Dijkstras algoritme/ <i>Dijkstra's algorithm</i>	118
Prims algoritme/ <i>Prim's algorithm</i>	123
Kruskals algoritme/ <i>Kruskal's algorithm</i>	128
Topologisk sortering/ <i>Topological sorting</i>	133
Stærke sammenhængskomponenter/ <i>Strongly connected components</i>	138
Løkke opgaver/ <i>Loop problems</i>	143
Amortiseret analyse/ <i>Amortized analysis</i>	152
Invarianter/ <i>Invariants</i>	161
Udvidede søgetræer/ <i>Augmented search trees</i>	173
Diverse spørgsmål/ <i>Miscellaneous questions</i>	183

Asymptotisk notation/ *Asymptotic notation*

Opgave/ *Problem 1* (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/ <i>Yes</i>	Nej/ <i>No</i>	
$(\log n)^3 = O(n^2)$	☒	☐	1.1
$4 \log(n^2) = O(n^{3/2})$	☒	☐	1.2
$n^2 + 3 \log n = O(\log n)$	☐	☒	1.3
$n^2 = O(n)$	☐	☒	1.4
$2n \cdot \log n = O(2^n)$	☒	☐	1.5
$3^3 + \sqrt{n} = O(n!)$	☒	☐	1.6
$3\sqrt{n} + 7n^{1/3} = O(n^{0.1})$	☐	☒	1.7
$4(\log n)^4 = O(2^n)$	☒	☐	1.8
$n \cdot \log n = O(\sqrt{n})$	☐	☒	1.9
$\log n = \Omega(\sqrt{n})$	☐	☒	1.10
$n \cdot \log n = \Omega(n^{0.1})$	☒	☐	1.11
$n^{0.01}/5 = \Omega(\log n)$	☒	☐	1.12

Opgave/ *Problem 2* (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/ <i>Yes</i>	Nej/ <i>No</i>	
$6\sqrt{n} \cdot \log n = O(n^{2/3})$	☒	☐	2.1
$n^{0.1} = O(n)$	☒	☐	2.2
$7(\log n)^2 = O(n^{0.001})$	☒	☐	2.3
$n \cdot \log n = O(\sqrt{n} \cdot \log n)$	☐	☒	2.4
$2^n = O(n^3)$	☐	☒	2.5
$n^2 \log n = O(\log n)$	☐	☒	2.6
$n^3 = O(\sqrt{n})$	☐	☒	2.7
$n = O(\log(n!))$	☒	☐	2.8
$7n \cdot \log n = O(n^{0.001})$	☐	☒	2.9
$n \cdot \log n = \Omega(8^{\log n})$	☐	☒	2.10
$\sqrt{n} = \Omega((\log n)^2)$	☒	☐	2.11
$n \cdot \log n = \Omega(n^{2/3})$	☒	☐	2.12

Opgave/Problem 3 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$\sqrt{n} = O(n^{2/3})$	☒	☐	3.1
$\log(n^2) = O(n \cdot \log n)$	☒	☐	3.2
$n = O(n^{0.01})$	☐	☒	3.3
$4n = O(n)$	☒	☐	3.4
$8^{\log n} = O(\log n)$	☐	☒	3.5
$2^n = O(\log(n^2))$	☐	☒	3.6
$\log(n!) = O(n)$	☐	☒	3.7
$n^2 = O(n^{0.001})$	☐	☒	3.8
$(\log n)^2 = O(n^{0.01})$	☒	☐	3.9
$n^{0.001} = \Omega(\log(n^2))$	☒	☐	3.10
$n^2 \log n = \Omega(\sqrt{n})$	☒	☐	3.11
$3^n = \Omega(\log n)$	☒	☐	3.12

Opgave/Problem 4 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$n + \sqrt{n} = O(n \cdot \log n)$	☒	☐	4.1
$2^{3 \log n} = O(n^n)$	☒	☐	4.2
$7n^3 = O(3^3)$	☐	☒	4.3
$n^2 \log n = O(\sqrt{n})$	☐	☒	4.4
$n \cdot \log n = O(n^3)$	☒	☐	4.5
$\log(n!) = O(n^3)$	☒	☐	4.6
$2^n = O(2^{3 \log n})$	☐	☒	4.7
$5n^{2/3} \cdot n^{1/3} = O(n^n)$	☒	☐	4.8
$\sqrt{n} = O(2^{\log n})$	☒	☐	4.9
$\log n + \log n = \Theta(\log n)$	☒	☐	4.10
$4\sqrt{n} \cdot \log n = \Omega(\sum_{i=1}^n i)$	☐	☒	4.11
$(\log n)^3 = \Theta((\log n)^6)$	☐	☒	4.12

Opgave/Problem 5 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$\log(n!) = O(n!)$	☒	☐ B	5.1
$2^n = O(n^{0.1})$	☐ A	☒	5.2
$n^{3/2} = O(n^2 \log n)$	☒	☐ B	5.3
$6n^2 = O(2^{\log n})$	☐ A	☒	5.4
$\log n = O(\sqrt{n} \cdot \log n)$	☒	☐ B	5.5
$5 \log n + n^2 = O(\log(n^2))$	☐ A	☒	5.6
$2^{3 \log n} = O(n)$	☐ A	☒	5.7
$1 = O(n^{0.1})$	☒	☐ B	5.8
$n \cdot \log n = O(n)$	☐ A	☒	5.9
$1 = \Omega(n^{1/3})$	☐ A	☒	5.10
$6 \log n = \Theta(4^{\log n})$	☐ A	☒	5.11
$n^{0.001} = \Omega(\sqrt{n} \cdot \log n)$	☐ A	☒	5.12

Opgave/Problem 6 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$\log n = O(2^{\log n})$	☒	☐ B	6.1
$8^{\log n} = O(n^{2/3})$	☐ A	☒	6.2
$\sqrt{n} = O(n^{2/3})$	☒	☐ B	6.3
$2^{\log n} = O(2^{2 \log n})$	☒	☐ B	6.4
$\log(n^2) = O(\log n)$	☒	☐ B	6.5
$\sqrt{n} \cdot \log n = O(\sqrt{n})$	☐ A	☒	6.6
$\log(n!)/7 = O(n^2 \log n)$	☒	☐ B	6.7
$n! + \sqrt{n} \cdot (\log n)/3 = O(\sqrt{n})$	☐ A	☒	6.8
$\sqrt{n} = O((\log n)^3)$	☐ A	☒	6.9
$5\sqrt{n} + 2 \cdot 8^{\log n} = \Omega(n!)$	☐ A	☒	6.10
$8^{\log n} = \Theta(2^{3 \log n})$	☒	☐ B	6.11
$n \cdot \log n = \Omega(2^{3 \log n})$	☐ A	☒	6.12

Opgave/Problem 7 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$1 + \log(n^2)/2 = O(\log n)$	☒	☐	7.1
$2n = O(n \cdot \log n)$	☒	☐	7.2
$1 + 3^3 = O(2^{3 \log n})$	☒	☐	7.3
$7n \cdot \log n = O(\log n)$	☐	☒	7.4
$n! = O((\log n)^3)$	☐	☒	7.5
$n^2 \log n = O(\sqrt{n})$	☐	☒	7.6
$n \cdot \log n = O(\log(n^2))$	☐	☒	7.7
$n^{2/3} = O(2^{3 \log n})$	☒	☐	7.8
$3n \cdot \log n = O(1)$	☐	☒	7.9
$n = \Theta(n^{0.1})$	☐	☒	7.10
$\log n = \Omega(n \cdot \log n)$	☐	☒	7.11
$4 \log n = \Theta(\log n)$	☒	☐	7.12

Opgave/Problem 8 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$n \cdot \log n = O(\sqrt{n})$	☐	☒	8.1
$n^{0.01} = O(\sqrt{n} \cdot \log n)$	☒	☐	8.2
$4 \cdot 3^n = O(\log n)$	☐	☒	8.3
$n^{2/3} = O(n^{2/3} \cdot n^{1/3})$	☒	☐	8.4
$5^5 + n^{0.001} = O(n \cdot \log n)$	☒	☐	8.5
$\log(n^2) = O(\log n)$	☒	☐	8.6
$5n^{1/3} = O(\log n)$	☐	☒	8.7
$2^{3 \log n} = O(n^3)$	☒	☐	8.8
$6\sqrt{n} \cdot \log n = O(\sqrt{n})$	☐	☒	8.9
$n/6 = \Theta(n)$	☒	☐	8.10
$2^{2 \log n} = \Theta(n\sqrt{n})$	☐	☒	8.11
$n^3 = \Theta(n \cdot \log n)$	☐	☒	8.12

Opgave/Problem 9 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$\sqrt{n}/2 = O(\log(n^2))$	☐ A	☒ B	9.1
$2^n = O(n \cdot \log n)$	☐ A	☒ B	9.2
$6 \log(n!) + n \cdot \log n = O(2^{\log n})$	☐ A	☒ B	9.3
$\log n = O(\sqrt{n} \cdot \log n)$	☒ A	☐ B	9.4
$2 = O((\log n)^2)$	☒ A	☐ B	9.5
$(\log n)^3 = O(n \cdot \log n)$	☒ A	☐ B	9.6
$n \cdot \log n = O(2^n)$	☒ A	☐ B	9.7
$n^3 + n \cdot \log n = O(n \cdot \log n)$	☐ A	☒ B	9.8
$8^{\log n} = O(2^n)$	☒ A	☐ B	9.9
$n^{0.1}/6 = \Theta(n^n)$	☐ A	☒ B	9.10
$4^{\log n} = \Theta(\sum_{i=1}^n i)$	☒ A	☐ B	9.11
$1 = \Theta(5^5)$	☒ A	☐ B	9.12

Opgave/Problem 10 (6 %)

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

In the following, $\log n$ denotes the base-2 logarithm of n .

	Ja/Yes	Nej/No	
$n \cdot \log n = O(2^{2 \log n})$	☒ A	☐ B	10.1
$2^{3 \log n} = O(n!)$	☒ A	☐ B	10.2
$n = O(n \cdot \log n)$	☒ A	☐ B	10.3
$n = O(n\sqrt{n})$	☒ A	☐ B	10.4
$3n = O(n)$	☒ A	☐ B	10.5
$\sqrt{n} + \sqrt{n} = O(n^{0.1})$	☐ A	☒ B	10.6
$\log(n!) + 6n^{3/2} = O(\log n)$	☐ A	☒ B	10.7
$n^{0.1} = O(\sqrt{n} \cdot \log n)$	☒ A	☐ B	10.8
$n \cdot (\log n)/3 = O(2^n)$	☒ A	☐ B	10.9
$n \cdot \log n = \Theta(\log(n!))$	☒ A	☐ B	10.10
$1 + 4 = \Theta(4^4)$	☒ A	☐ B	10.11
$\sqrt{n} \cdot \log n = \Omega(\sqrt{n})$	☒ A	☐ B	10.12

Analyse af løkker / *Analysis of loops*

Opgave / *Problem* 11 (6 %)

Algorithm loop1(n) $s = 1$ while $s \leq n$ $s = s + 1$	Algorithm loop2(n) $s = 0$ for $i = 1$ to n for $j = 1$ to n for $k = 1$ to n $s = s + 1$
Algorithm loop3(n) $i = 0$ $j = n$ while $i < j$ $i = i + 2$ $j = j + 1$	Algorithm loop4(n) for $i = 1$ to n $j = i$ while $j \leq n$ $j = 2 * j$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta((\log n)^2)$	$\Theta(n^3)$	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(\log \log n)$	$\Theta(\log n)$	$\Theta(n^2)$	$\Theta(n^2 \cdot \log n)$	
loop1	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	11.1
loop2	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	11.2
loop3	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	11.3
loop4	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	11.4

Opgave/Problem 12 (6 %)

Algorithm loop1(n) $i = 1$ while $i \leq n$ $i = 2 * i$	Algorithm loop2(n) $s = 1$ for $i = 1$ to n for $j = i$ to n $s = s + 1$
Algorithm loop3(n) $i = 1$ $j = n$ while $i \leq j$ $i = i * 2$ $j = \lfloor j/2 \rfloor$	Algorithm loop4(n) for $i = 1$ to n $j = 0$ while $j \leq n$ $j = j + i$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(\log n)$	$\Theta((\log n)^2)$	$\Theta(\log \log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^3)$	$\Theta(n^2)$	$\Theta(n^2 \cdot \log n)$	
loop1	☒	☐	☐	☐	☐	☐	☐	☐	12.1
loop2	☐	☐	☐	☐	☐	☐	☒	☐	12.2
loop3	☒	☐	☐	☐	☐	☐	☐	☐	12.3
loop4	☐	☐	☐	☐	☒	☐	☐	☐	12.4

Opgave/Problem 13 (6 %)

Algorithm loop1(n)	Algorithm loop2(n)
$s = 0$	$i = 1$
for $i = 1$ to n	while $i \leq n$
for $j = 1$ to $i * i$	$i = 2 * i$
$s = s + 1$	
 Algorithm loop3(n)	 Algorithm loop4(n)
$i = 1$	$i = 1$
while $i \leq n * n$	while $i \leq n$
$i = 2 * i$	$j = 1$
	while $j \leq i$
	$j = 2 * j$
	$i = 2 * i$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2)$	$\Theta((\log n)^2)$	$\Theta(\log n)$	$\Theta(n^3)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(\sqrt{n})$	$\Theta(n^2 \cdot \log n)$	
loop1	A	B	C	D	E	F	G	H	13.1
loop2	A	B	C	D	E	F	G	H	13.2
loop3	A	B	C	D	E	F	G	H	13.3
loop4	A	B	C	D	E	F	G	H	13.4

Opgave/Problem 14 (6 %)

Algorithm loop1(n) **Algorithm** loop2(n)

$i = 1$

$i = n$

while $i \leq n$

while $i > 0$

$i = 2 * i$

$i = i - 1$

Algorithm loop3(n) **Algorithm** loop4(n)

$i = 1$

$i = 1$

while $i \leq n * n$

$j = 1$

$i = 3 * i$

$s = 0$

while $s \leq n$

while $j \leq s$

$j = 2 * j$

$s = s + i$

$i = i + 1$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2)$	$\Theta(2^n)$	$\Theta(\sqrt{n})$	$\Theta(n^3)$	$\Theta(\log n)$	$\Theta(n \log n)$	$\Theta(n)$	$\Theta((\log n)^2)$	
loop1	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	14.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	14.2
loop3	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	14.3
loop4	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	14.4

Opgave/Problem 15 (6 %)

Algorithm loop1(n) <pre> $s = 0$ for $i = 1$ to n for $j = 1$ to n for $k = 1$ to n $s = s + 1$ </pre>	Algorithm loop2(n) <pre> for $i = 1$ to n $j = i$ while $j > 0$ $j = j - 1$ </pre>
Algorithm loop3(n) <pre> $i = 1$ while $i \leq n$ $j = 0$ while $j \leq n$ $j = j + 1$ $i = 2 * i$ </pre>	Algorithm loop4(n) <pre> $i = n$ $j = 0$ while $i > 0$ if $j < i$ $j = j + 1$ else $j = 0$ $i = i - 1$ </pre>

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n \log n)$	$\Theta((\log n)^2)$	$\Theta(n)$	$\Theta(n^3)$	$\Theta(\sqrt{n})$	$\Theta(\frac{\log n}{\log \log n})$	$\Theta(n^2)$	$\Theta(\log n)$	
loop1	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	15.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	15.2
loop3	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	15.3
loop4	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	15.4

Opgave/Problem 16 (6 %)

Algorithm loop1(n) **Algorithm** loop2(n)

```

 $i = 1$ 
while  $i \leq n$ 
     $j = 1$ 
    while  $j \leq n$ 
         $j = j + 1$ 
     $i = i + 1$ 

```

```

 $s = 0$ 
for  $i = 1$  to  $n$ 
    for  $j = 1$  to  $i * i$ 
         $s = s + 1$ 

```

Algorithm loop3(n) **Algorithm** loop4(n)

```

 $i = n$ 
while  $i \leq n * n$ 
     $i = 2 * i$ 

```

```

 $i = 1$ 
while  $i \leq n$ 
     $j = i$ 
    while  $j \leq n$ 
         $j = j + 1$ 
     $i = 2 * i$ 

```

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n \log n)$	$\Theta(\log \log n)$	$\Theta(n^3)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta((\log n)^2)$	$\Theta(n^2)$	$\Theta(n\sqrt{n})$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	16.1
loop2	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	16.2
loop3	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	16.3
loop4	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	16.4

Opgave/Problem 17 (6 %)

Algorithm loop1(n) <pre> $s = 1$ for $i = 1$ to $n * n$ for $j = 1$ to n $s = s + 1$ </pre>	Algorithm loop2(n) <pre> $s = 1$ for $i = 1$ to n $s = s + 1$ </pre>
Algorithm loop3(n) <pre> $s = 0$ $i = n$ while $i > 0$ for $j = 1$ to i $s = s + 1$ $i = i - 1$ </pre>	Algorithm loop4(n) <pre> $i = n$ $j = 0$ while $i > 0$ if $j < i$ $j = j + 1$ else $j = 0$ $i = i - 1$ </pre>

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(\sqrt{n} \log n)$	$\Theta((\log n)^2)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n)$	$\Theta(\log n)$	$\Theta(\sqrt{n})$	
loop1	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	17.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	☐ H	17.2
loop3	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	17.3
loop4	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	17.4

Opgave/Problem 18 (6 %)

Algorithm loop1(n) $s = 0$ for $i = 1$ to n for $j = 1$ to n for $k = 1$ to n $s = s + 1$	Algorithm loop2(n) $i = n$ while $i > 0$ $i = i - 1$
Algorithm loop3(n) $i = 2$ while $i \leq n$ $i = i * i$	Algorithm loop4(n) for $i = 1$ to n $j = i$ while $j \leq n$ $j = 2 * j$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2)$	$\Theta(n)$	$\Theta(\log \log n)$	$\Theta(\sqrt{n})$	$\Theta((\log n)^2)$	$\Theta(n \log n)$	$\Theta(n^3)$	$\Theta(\log n)$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	18.1
loop2	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	18.2
loop3	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	18.3
loop4	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	18.4

Opgave/Problem 19 (6 %)

Algorithm loop1(n) for $i = 1$ to n $j = i$ while $j > 0$ $j = j - 1$	Algorithm loop2(n) $i = 1$ while $i \leq n$ $i = 2 * i$
Algorithm loop3(n) $s = 0$ $i = 1$ while $s \leq n$ $s = s + i$ $i = i + 1$	Algorithm loop4(n) $i = 1$ while $i \leq n$ $j = i$ while $j \leq n$ $j = j + 1$ $i = 2 * i$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta((\log n)^2)$	$\Theta(\sqrt{n})$	$\Theta(n^2 \cdot \log n)$	$\Theta(n^3)$	$\Theta(n^2)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	
loop1	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	19.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	☐ H	19.2
loop3	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	19.3
loop4	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	19.4

Opgave/Problem 20 (6 %)

Algorithm loop1(n) for $i = 1$ to n $j = i$ while $j > 0$ $j = j - 1$	Algorithm loop2(n) $s = 1$ for $i = n$ to 1 step -1 for $j = n$ to 1 step -1 $s = s + 1$
Algorithm loop3(n) $i = 1$ while $i \leq n$ $j = 1$ while $j \leq i$ $j = 2 * j$ $i = i + 1$	Algorithm loop4(n) $i = 1$ $j = n$ while $i \leq j$ $i = 4 * i$ $j = 2 * j$

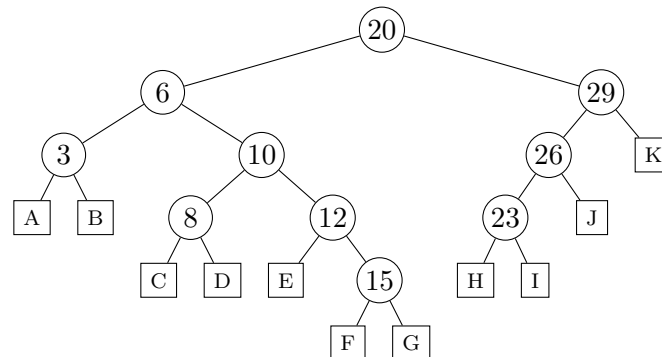
Angiv for hver af ovenstående algoritmer udførselstiden som funktion af n i Θ -notation.

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n)$	$\Theta(n^3)$	$\Theta(\log n)$	$\Theta(\frac{\log n}{\log \log n})$	$\Theta(2^n)$	$\Theta((\log n)^2)$	$\Theta(n \log n)$	$\Theta(n^2)$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	20.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	20.2
loop3	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	20.3
loop4	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	20.4

Indsættelse i søgetræer/*Insertions into search trees*

Opgave/*Problem* 21 (4 %)

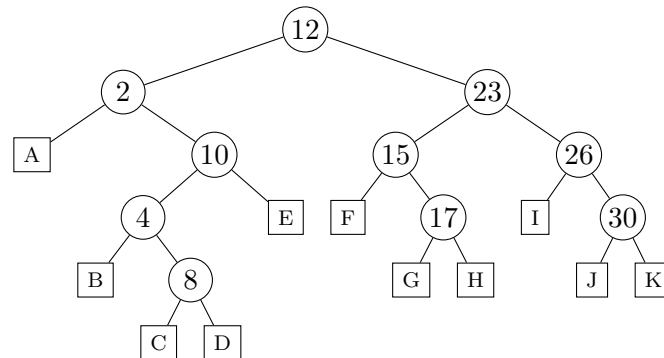


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 18, 2, 11, 25 and 22 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 18, 2, 11, 25 and 22 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(18)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	21.1
INSERT(2)	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	21.2
INSERT(11)	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	21.3
INSERT(25)	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	21.4
INSERT(22)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	21.5

Opgave/Problem 22 (4%)

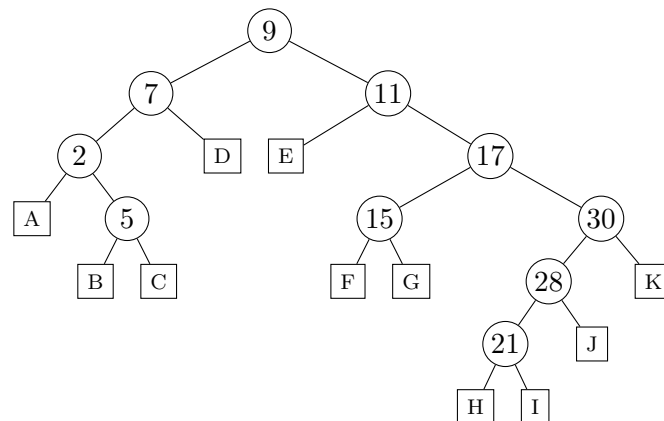


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 19, 28, 27, 1 and 21 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 19, 28, 27, 1 and 21 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(19)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	22.1
INSERT(28)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	22.2
INSERT(27)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	22.3
INSERT(1)	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	22.4
INSERT(21)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	22.5

Opgave/Problem 23 (4%)

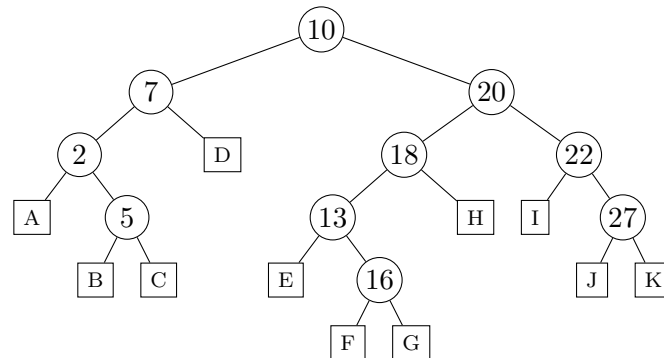


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 10, 3, 22, 25 and 20 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 10, 3, 22, 25 and 20 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(10)	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	23.1
INSERT(3)	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	23.2
INSERT(22)	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	23.3
INSERT(25)	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	23.4
INSERT(20)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	23.5

Opgave/Problem 24 (4 %)

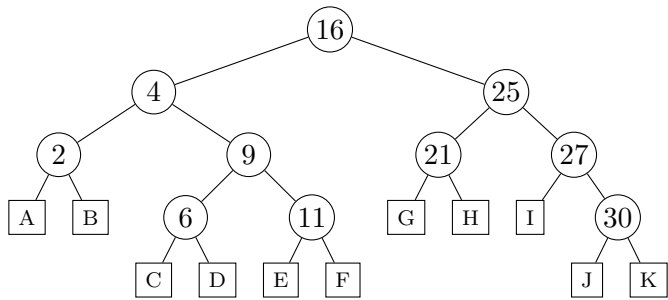


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 11, 19, 17, 28 og 1 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 11, 19, 17, 28 and 1 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(11)	A	B	C	D	E	F	G	H	I	J	K	24.1
INSERT(19)	A	B	C	D	E	F	G	H	I	J	K	24.2
INSERT(17)	A	B	C	D	E	F	G	H	I	J	K	24.3
INSERT(28)	A	B	C	D	E	F	G	H	I	J	K	24.4
INSERT(1)	A	B	C	D	E	F	G	H	I	J	K	24.5

Opgave/Problem 25 (4%)

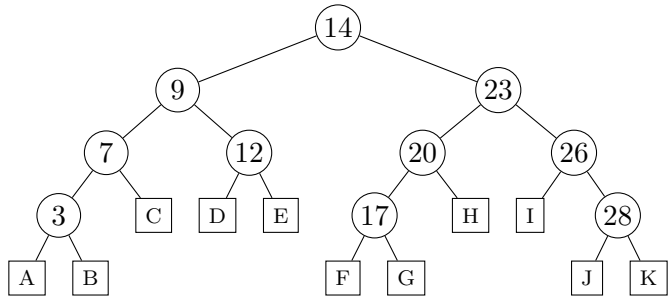


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 18, 5, 12, 19 and 22 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 18, 5, 12, 19 and 22 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(18)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	25.1
INSERT(5)	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	25.2
INSERT(12)	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	25.3
INSERT(19)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	25.4
INSERT(22)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	25.5

Opgave/Problem 26 (4 %)

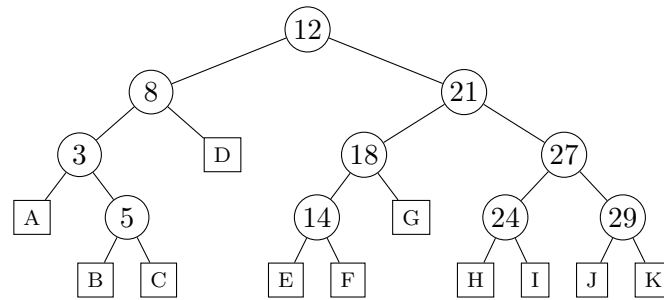


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 16, 2, 4, 19 and 22 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 16, 2, 4, 19 and 22 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(16)	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	26.1
INSERT(2)	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	26.2
INSERT(4)	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	26.3
INSERT(19)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	26.4
INSERT(22)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	26.5

Opgave/Problem 27 (4%)

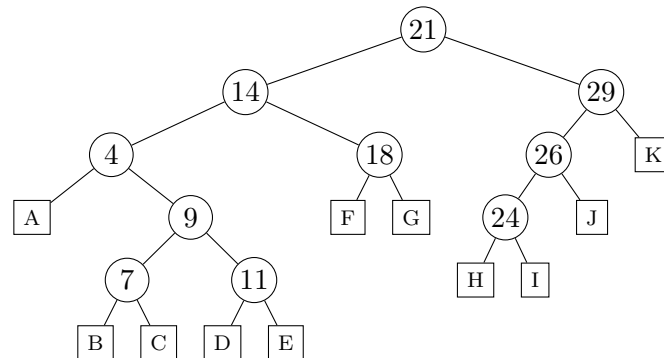


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 17, 10, 19, 15 and 23 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 17, 10, 19, 15 and 23 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(17)	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	27.1
INSERT(10)	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	27.2
INSERT(19)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	27.3
INSERT(15)	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	27.4
INSERT(23)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	27.5

Opgave/Problem 28 (4 %)

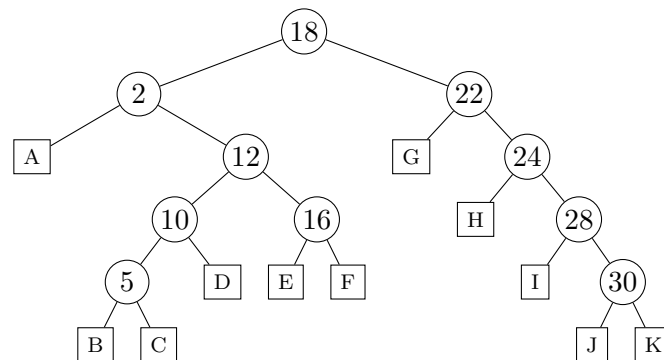


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 15, 20, 28, 16 and 12 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 15, 20, 28, 16 and 12 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(15)	A	B	C	D	E	F	G	H	I	J	K	28.1
INSERT(20)	A	B	C	D	E	F	G	H	I	J	K	28.2
INSERT(28)	A	B	C	D	E	F	G	H	I	J	K	28.3
INSERT(16)	A	B	C	D	E	F	G	H	I	J	K	28.4
INSERT(12)	A	B	C	D	E	F	G	H	I	J	K	28.5

Opgave/Problem 29 (4 %)

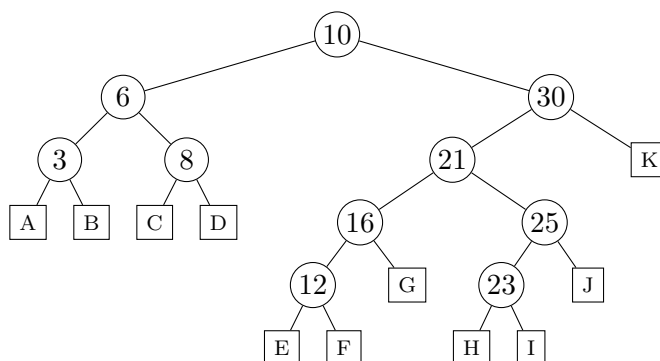


Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 26, 29, 21, 8 og 6 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 26, 29, 21, 8 and 6 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(26)	A	B	C	D	E	F	G	H	X	J	K	29.1
INSERT(29)	A	B	C	D	E	F	G	H	I	X	K	29.2
INSERT(21)	A	B	C	D	E	F	X	H	I	J	K	29.3
INSERT(8)	A	B	X	D	E	F	G	H	I	J	K	29.4
INSERT(6)	A	B	X	D	E	F	G	H	I	J	K	29.5

Opgave/Problem 30 (4%)



Angiv i hvilke blade A–K i ovenstående ubalancerede binære søgetræ elementerne 18, 7, 4, 24 and 9 skal indsættes (det antages at før hver indsættelse indeholder træet kun ovenstående ti elementer).

Indicate in which leaves A–K in the above unbalanced binary search tree the elements 18, 7, 4, 24 and 9 should be inserted (it is assumed that before each insertion the tree contains only the above ten elements).

	A	B	C	D	E	F	G	H	I	J	K	
INSERT(18)	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	30.1
INSERT(7)	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	30.2
INSERT(4)	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	30.3
INSERT(24)	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐	30.4
INSERT(9)	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	30.5

Max-Heap-Insert

Opgave/Problem 31 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 3, 11, 13, 14, 1, 8 og 5 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 3, 11, 13, 14, 1, 8 and 5 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
14	13	11	3	1	8	5	☒
1	2	3	4	5	6	7	
14	11	13	3	1	8	5	☐
1	2	3	4	5	6	7	
3	11	13	14	1	8	5	☐
1	2	3	4	5	6	7	
14	13	11	8	5	3	1	☐
1	2	3	4	5	6	7	
13	14	8	11	1	3	5	☐

Opgave/Problem 32 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 2, 7, 4, 9, 8, 11 og 3 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 2, 7, 4, 9, 8, 11 and 3 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
11	8	9	2	7	4	3	☒

1	2	3	4	5	6	7	
11	9	8	7	4	3	2	☐

1	2	3	4	5	6	7	
7	9	11	2	8	4	3	☐

1	2	3	4	5	6	7	
11	9	4	7	8	2	3	☐

1	2	3	4	5	6	7	
2	7	4	9	8	11	3	☐

Opgave/Problem 33 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 10, 2, 1, 13, 12, 14 og 6 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 10, 2, 1, 13, 12, 14 and 6 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
14	13	10	2	12	1	6	☐

1	2	3	4	5	6	7	
14	12	13	2	10	1	6	☒

1	2	3	4	5	6	7	
14	13	12	10	6	2	1	☐

1	2	3	4	5	6	7	
10	13	14	2	12	1	6	☐

1	2	3	4	5	6	7	
10	2	1	13	12	14	6	☐

Opgave/Problem 34 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 10, 12, 1, 14, 6, 11 og 13 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 10, 12, 1, 14, 6, 11 and 13 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

- | | | | | | | |
|----|----|---|----|---|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 12 | 1 | 14 | 6 | 11 | 13 |
- ☐
- A
- | | | | | | | |
|----|----|----|----|----|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 13 | 12 | 11 | 10 | 6 | 1 |
- ☐
- B
- | | | | | | | |
|----|----|----|----|---|----|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 12 | 14 | 13 | 10 | 6 | 11 | 1 |
- ☐
- C
- | | | | | | | |
|----|----|----|----|---|---|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 12 | 13 | 10 | 6 | 1 | 11 |
- ☒
- D
- | | | | | | | |
|----|----|----|----|---|----|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 12 | 13 | 10 | 6 | 11 | 1 |
- ☐
- E

Opgave/Problem 35 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 4, 10, 1, 2, 5, 12 og 3 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 4, 10, 1, 2, 5, 12 and 3 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

- | | | | | | | |
|----|----|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 12 | 10 | 5 | 4 | 3 | 2 | 1 |
- ☐
- A
- | | | | | | | |
|----|----|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 12 | 10 | 4 | 2 | 5 | 1 | 3 |
- ☐
- B
- | | | | | | | |
|----|---|----|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 12 | 5 | 10 | 2 | 4 | 1 | 3 |
- ☒
- C
- | | | | | | | |
|----|---|----|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 5 | 12 | 2 | 4 | 1 | 3 |
- ☐
- D
- | | | | | | | |
|---|----|---|---|---|----|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 4 | 10 | 1 | 2 | 5 | 12 | 3 |
- ☐
- E

Opgave/Problem 36 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 6, 3, 9, 11, 12, 8 og 14 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 6, 3, 9, 11, 12, 8 and 14 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
6	3	9	11	12	8	14	☐ A

1	2	3	4	5	6	7	
9	12	14	11	3	8	6	☐ B

1	2	3	4	5	6	7	
14	12	11	9	8	6	3	☐ C

1	2	3	4	5	6	7	
14	11	12	3	9	6	8	☒

1	2	3	4	5	6	7	
14	12	9	11	3	8	6	☐ E

Opgave/Problem 37 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 3, 12, 4, 10, 11, 2 og 14 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 3, 12, 4, 10, 11, 2 and 14 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
14	12	4	10	11	2	3	☐ A

1	2	3	4	5	6	7	
3	12	4	10	11	2	14	☐ B

1	2	3	4	5	6	7	
14	11	12	3	10	2	4	☒

1	2	3	4	5	6	7	
12	11	14	10	3	2	4	☐ D

1	2	3	4	5	6	7	
14	12	11	10	4	3	2	☐ E

Opgave/Problem 38 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 6, 8, 10, 14, 7, 1 og 12 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 6, 8, 10, 14, 7, 1 and 12 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

- | | | | | | | |
|----|---|----|---|---|---|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 8 | 12 | 6 | 7 | 1 | 10 |
- ☐
- A
- | | | | | | | |
|----|----|----|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 12 | 10 | 8 | 7 | 6 | 1 |
- ☐
- B
- | | | | | | | |
|----|----|----|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 14 | 12 | 8 | 7 | 1 | 6 |
- ☐
- C
- | | | | | | | |
|----|----|----|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 14 | 10 | 12 | 6 | 7 | 1 | 8 |
- ☒
- D
- | | | | | | | |
|---|---|----|----|---|---|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 6 | 8 | 10 | 14 | 7 | 1 | 12 |
- ☐
- E

Opgave/Problem 39 (4%)

Angiv den binære max-heap efter indsættelse af elementerne 2, 8, 1, 5, 10, 9 og 6 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 2, 8, 1, 5, 10, 9 and 6 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

- | | | | | | | |
|----|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 8 | 9 | 2 | 5 | 1 | 6 |
- ☒
- A
- | | | | | | | |
|----|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 8 | 9 | 5 | 2 | 1 | 6 |
- ☐
- B
- | | | | | | | |
|---|----|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 10 | 9 | 5 | 2 | 1 | 6 |
- ☐
- C
- | | | | | | | |
|---|---|---|---|----|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 8 | 1 | 5 | 10 | 9 | 6 |
- ☐
- D
- | | | | | | | |
|----|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 9 | 8 | 6 | 5 | 2 | 1 |
- ☐
- E

Opgave/Problem 40 (4 %)

Angiv den binære max-heap efter indsættelse af elementerne 10, 5, 6, 9, 8, 11 og 14 i den givne rækkefølge med MAX-HEAP-INSERT, startende med den tomme heap.

What is the binary max-heap after inserting the elements 10, 5, 6, 9, 8, 11 and 14 in the given order with MAX-HEAP-INSERT, starting with the empty heap.

1	2	3	4	5	6	7	
10	9	14	5	8	11	6	☐ A

1	2	3	4	5	6	7	
14	9	11	5	8	6	10	☒ B

1	2	3	4	5	6	7	
10	5	6	9	8	11	14	☐ C

1	2	3	4	5	6	7	
14	9	11	5	8	10	6	☐ D

1	2	3	4	5	6	7	
14	11	10	9	8	6	5	☐ E

Build-Max-Heap

Opgave/Problem 41 (4 %)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

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

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

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

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

Opgave/Problem 42 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☒

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

☐

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

☐

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

☐

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

☐

Opgave/Problem 43 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐

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

☒

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

☐

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

☐

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

☐

Opgave/Problem 44 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ 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	6	5	7	2	1	3	4

☒

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

☐ E

Opgave/Problem 45 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☐ C

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

☒

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

☐ E

Opgave/Problem 46 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☐ C

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

☐ D

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

☒

Opgave/Problem 47 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☒

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

☐ D

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

☐ E

Opgave/Problem 48 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☐ C

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

☒ D

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

☐ E

Opgave/Problem 49 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☐ C

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

☒ D

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

☐ E

Opgave/Problem 50 (4%)

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

Hvad er resultatet af at anvende BUILD-MAX-HEAP på ovenstående array?

What is the result of applying BUILD-MAX-HEAP on the above array?

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

☐ A

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

☐ B

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

☐ C

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

☒

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

☐ E

Heap-Extract-Max

Opgave/Problem 51 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	19	18	23	10	16	14	6	1	7	4	8

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

1	2	3	4	5	6	7	8	9	10	11	12
24	23	19	18	8	10	16	14	6	1	7	4

☒

1	2	3	4	5	6	7	8	9	10	11	12	13
24	23	19	18	7	10	16	14	6	1		4	8

☐ B

1	2	3	4	5	6	7	8	9	10	11	12
24	23	19	18	7	10	16	14	6	1	8	4

☐ C

1	2	3	4	5	6	7	8	9	10	11	12
24	23	19	18	7	10	16	14	6	1	4	8

☐ D

1	2	3	4	5	6	7	8	9	10	11	12
24	23	18	19	10	16	14	6	1	7	4	8

☐ E

Opgave/Problem 52 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	23	8	16	18	9	4	3	7	10	13	15

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 16 | 23 | 8 | 10 | 18 | 9 | 4 | 3 | 7 | 15 | 13 | ☐ |
- A**
- | | | | | | | | | | | | | |
|----|----|----|----|----|---|---|---|---|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 23 | 15 | 16 | 18 | 9 | 4 | 3 | 7 | 10 | 13 | 8 | ☐ |
- B**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 16 | 23 | 8 | 10 | 18 | 9 | 4 | 3 | 7 | 13 | 15 | ☐ |
- C**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 24 | 16 | 23 | 8 | 10 | 18 | 9 | 4 | 3 | 7 | | 13 | 15 |
- D**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|-------------------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 16 | 23 | 8 | 15 | 18 | 9 | 4 | 3 | 7 | 10 | 13 | ☒ |
- X**

Opgave/Problem 53 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	22	19	17	13	20	7	6	11	5	9	8

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | | | | | | | | | | | | | |
|----|----|----|---|----|----|----|---|---|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 19 | 22 | 7 | 17 | 13 | 20 | 8 | 6 | 11 | 5 | 9 | ☐ |
- A**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|----|---|---|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 24 | 19 | 22 | 7 | 17 | 13 | 20 | | 6 | 11 | 5 | 9 | 8 |
- B**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|----|---|---|----|----|----|-------------------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 19 | 22 | 8 | 17 | 13 | 20 | 7 | 6 | 11 | 5 | 9 | ☒ |
- X**
- | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|---|----|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 22 | 20 | 17 | 13 | 19 | 7 | 6 | 11 | 5 | 9 | 8 | ☐ |
- D**
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|----|---|----|----|----|----|--------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 19 | 22 | 7 | 17 | 13 | 20 | 6 | 11 | 5 | 9 | 8 | ☐ |
- E**

Opgave/Problem 54 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	18	16	13	12	15	8	5	4	7	1	14	6

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 18 | 13 | 16 | 5 | 12 | 15 | 8 | 6 | 4 | 7 | 1 | 14 | ☐ A |
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 18 | 13 | 16 | 5 | 12 | 15 | 8 | 4 | 7 | 1 | 14 | 6 | ☐ B |
- | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|-------------------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 18 | 13 | 16 | 6 | 12 | 15 | 8 | 5 | 4 | 7 | 1 | 14 | ☒ |
- | | | | | | | | | | | | | | |
|----|----|----|---|----|----|---|---|---|----|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
| 18 | 13 | 16 | 5 | 12 | 15 | 8 | | 4 | 7 | 1 | 14 | 6 | ☐ D |
- | | | | | | | | | | | | | |
|----|----|----|----|----|---|---|---|---|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 18 | 16 | 13 | 12 | 15 | 8 | 5 | 4 | 7 | 1 | 14 | 6 | ☐ E |

Opgave/Problem 55 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	22	21	19	20	5	1	16	14	11	3	18

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|----|----|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 22 | 21 | 19 | 20 | 18 | 1 | 16 | 14 | 11 | 3 | 5 | ☐ A |
- | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|---|----|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 21 | 22 | 16 | 19 | 20 | 5 | 1 | 14 | 11 | 3 | 18 | ☐ B |
- | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|---|----|----|----|----|-------------------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 21 | 22 | 18 | 19 | 20 | 5 | 1 | 16 | 14 | 11 | 3 | ☒ |
- | | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|---|---|----|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
| 24 | 21 | 22 | 16 | 19 | 20 | 5 | 1 | | 14 | 11 | 3 | 18 | ☐ D |
- | | | | | | | | | | | | | |
|----|----|----|----|----|----|---|---|----|----|----|----|----------------------------|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 24 | 21 | 22 | 16 | 19 | 20 | 5 | 1 | 18 | 14 | 11 | 3 | ☐ E |

Opgave/Problem 56 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	18	22	19	13	4	7	9	17	12	6	10

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|---|----|----|---|---|----|----|----|----|----------------------------|
| 24 | 22 | 18 | 9 | 19 | 13 | 4 | 7 | 17 | 12 | 6 | 10 | ☐ A |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|----|----|----|---|---|---|----|----|----|-------------------------------------|
| 24 | 22 | 18 | 10 | 19 | 13 | 4 | 7 | 9 | 17 | 12 | 6 | ☒ |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|----|----|----|---|---|----|----|----|----|----------------------------|
| 24 | 19 | 22 | 18 | 13 | 10 | 7 | 9 | 17 | 12 | 6 | 4 | ☐ C |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|---|----|----|---|---|----|----|----|----|----------------------------|
| 24 | 22 | 18 | 9 | 19 | 13 | 4 | 7 | 10 | 17 | 12 | 6 | ☐ D |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
|----|----|----|---|----|----|---|---|---|----|----|----|----|----------------------------|
| 24 | 22 | 18 | 9 | 19 | 13 | 4 | 7 | | 17 | 12 | 6 | 10 | ☐ E |

Opgave/Problem 57 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
25	24	21	18	14	19	11	8	4	9	6	2	13

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|----|----|----|---|---|---|----|----|----|----------------------------|
| 24 | 21 | 18 | 14 | 19 | 13 | 8 | 4 | 9 | 6 | 2 | 11 | ☐ A |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|----|----|----|----|---|---|----|----|----|-------------------------------------|
| 24 | 18 | 21 | 13 | 14 | 19 | 11 | 8 | 4 | 9 | 6 | 2 | ☒ |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|---|----|----|----|---|---|----|----|----|----------------------------|
| 24 | 18 | 21 | 8 | 14 | 19 | 11 | 4 | 9 | 6 | 2 | 13 | ☐ C |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|----|----|----|---|----|----|----|----|---|----|----|----|----------------------------|
| 24 | 18 | 21 | 8 | 14 | 19 | 11 | 13 | 4 | 9 | 6 | 2 | ☐ D |
- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | |
|----|----|----|---|----|----|----|---|---|----|----|----|----|----------------------------|
| 24 | 18 | 21 | 8 | 14 | 19 | 11 | | 4 | 9 | 6 | 2 | 13 | ☐ E |

Opgave/Problem 58 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
23	22	18	21	15	16	13	6	9	11	14	5	12

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

1	2	3	4	5	6	7	8	9	10	11	12	13	
22	21	18	9	15	16	13	6		11	14	5	12	☐

1	2	3	4	5	6	7	8	9	10	11	12		
22	21	18	9	15	16	13	6	12	11	14	5		☐

1	2	3	4	5	6	7	8	9	10	11	12		
22	18	21	15	16	13	6	9	11	14	5	12		☐

1	2	3	4	5	6	7	8	9	10	11	12		
22	21	18	9	15	16	13	6	11	14	5	12		☐

1	2	3	4	5	6	7	8	9	10	11	12		
22	21	18	12	15	16	13	6	9	11	14	5		☒

Opgave/Problem 59 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
26	25	22	19	10	13	20	8	7	5	1	6	9

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

1	2	3	4	5	6	7	8	9	10	11	12		
25	19	22	8	10	13	20	9	7	5	1	6		☐

1	2	3	4	5	6	7	8	9	10	11	12		
25	19	22	9	10	13	20	8	7	5	1	6		☒

1	2	3	4	5	6	7	8	9	10	11	12		
25	22	20	10	13	19	8	7	5	1	6	9		☐

1	2	3	4	5	6	7	8	9	10	11	12		
25	19	22	8	10	13	20	7	5	1	6	9		☐

1	2	3	4	5	6	7	8	9	10	11	12	13	
25	19	22	8	10	13	20		7	5	1	6	9	☐

Opgave/Problem 60 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13
26	24	22	13	23	19	8	11	12	2	10	4	15

Hvad er resultatet af at anvende HEAP-EXTRACT-MAX på ovenstående max-heap?

What is the result of applying HEAP-EXTRACT-MAX to the above max-heap?

1	2	3	4	5	6	7	8	9	10	11	12	13	
24	23	22	13	10	19	8	11	12	2		4	15	☐ A

1	2	3	4	5	6	7	8	9	10	11	12		
24	23	22	13	10	19	8	11	12	2	4	15		☐ B

1	2	3	4	5	6	7	8	9	10	11	12		
24	23	15	22	19	13	11	12	2	10	4	8		☐ C

1	2	3	4	5	6	7	8	9	10	11	12		
24	23	22	13	15	19	8	11	12	2	10	4		☒

1	2	3	4	5	6	7	8	9	10	11	12		
24	23	22	13	10	19	8	11	12	2	15	4		☐ E

Partition

Opgave/Problem 61 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
14	17	1	25	3	13	22	24	9	7	29	30	10	23	12

Angiv resultatet af at anvende PARTITION($A, 3, 14$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 3, 14$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
14	17	1	3	7	9	10	13	22	23	24	25	29	30	12	☐ A

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
14	17	1	3	13	22	9	7	10	23	29	30	25	24	12	☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
14	17	1	3	13	22	9	7	10	23	25	24	29	30	12	☐ C

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
1	3	7	9	10	12	13	14	17	22	23	24	25	29	30	☐ D

Opgave/Problem 62 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	21	26	23	28	18	22	19	8	4	20	6	24	12	3

Angiv resultatet af at anvende PARTITION($A, 2, 12$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 2, 12$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	4	6	23	28	18	22	19	8	21	20	26	24	12	3

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3	4	6	8	12	15	18	19	20	21	22	23	24	26	28

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	4	6	8	18	19	20	21	22	23	26	28	24	12	3

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	4	6	21	26	23	28	18	22	19	8	20	24	12	3

☐

Opgave/Problem 63 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	17	8	23	29	2	18	26	22	20	25	11	1	9	28

Angiv resultatet af at anvende PARTITION($A, 3, 13$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 3, 13$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2	8	9	11	17	18	20	21	22	23	25	26	28	29

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	17	1	8	23	29	2	18	26	22	20	25	11	9	28

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	17	1	2	8	11	18	20	22	23	25	26	29	9	28

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	17	1	23	29	2	18	26	22	20	25	11	8	9	28

☒

Opgave/Problem 64 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17	13	3	15	4	27	30	21	22	1	20	8	12	11	25

Angiv resultatet af at anvende PARTITION($A, 4, 12$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 4, 12$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17	13	3	4	1	8	30	21	22	15	20	27	12	11	25

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17	13	3	1	4	8	15	20	21	22	27	30	12	11	25

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17	13	3	4	1	8	15	27	30	21	22	20	12	11	25

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	3	4	8	11	12	13	15	17	20	21	22	25	27	30

☐

Opgave/Problem 65 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	24	30	25	27	3	17	29	4	15	23	2	6	5	10

Angiv resultatet af at anvende PARTITION($A, 4, 12$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 4, 12$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	24	30	2	3	4	15	17	23	25	27	29	6	5	10

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	24	30	2	27	3	17	29	4	15	23	25	6	5	10

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	4	5	6	10	15	17	21	23	24	25	27	29	30

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
21	24	30	2	25	27	3	17	29	4	15	23	6	5	10

☐

Opgave/Problem 66 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
30	12	26	1	5	19	8	28	29	9	22	18	2	27	21

Angiv resultatet af at anvende PARTITION($A, 4, 13$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 4, 13$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
30	12	26	1	2	19	8	28	29	9	22	18	5	27	21

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
30	12	26	1	2	5	8	9	18	19	22	28	29	27	21

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
30	12	26	1	2	5	19	8	28	29	9	22	18	27	21

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2	5	8	9	12	18	19	21	22	26	27	28	29	30

☐

Opgave/Problem 67 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
12	26	15	10	27	25	19	6	8	3	16	23	4	1	22

Angiv resultatet af at anvende PARTITION($A, 3, 14$) på ovenstående array A .

Indicate the result of applying PARTITION($A, 3, 14$) to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
12	26	1	10	27	25	19	6	8	3	16	23	4	15	22

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	3	4	6	8	10	12	15	16	19	22	23	25	26	27

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
12	26	1	15	10	27	25	19	6	8	3	16	23	4	22

☐

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
12	26	1	3	4	6	8	10	15	16	19	23	25	27	22

☐

Opgave/Problem 68 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
19	24	1	11	23	6	18	21	22	10	3	2	25	13	12

Angiv resultatet af at anvende $\text{PARTITION}(A, 4, 14)$ på ovenstående array A .

Indicate the result of applying $\text{PARTITION}(A, 4, 14)$ to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
19	24	1	11	6	10	3	2	13	23	18	21	22	25	12

☐ A

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
19	24	1	2	3	6	10	11	13	18	21	22	23	25	12

☐ B

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2	3	6	10	11	12	13	18	19	21	22	23	24	25

☐ C

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
19	24	1	11	6	10	3	2	13	23	18	21	25	22	12

☒

Opgave/Problem 69 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
7	27	11	15	21	23	10	12	1	5	16	19	13	18	28

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

Indicate the result of applying $\text{PARTITION}(A, 2, 13)$ to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
7	11	10	12	1	5	13	15	21	23	16	19	27	18	28

☒

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
7	11	10	12	1	5	13	27	15	21	23	16	19	18	28

☐ B

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	5	7	10	11	12	13	15	16	18	19	21	23	27	28

☐ C

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
7	1	5	10	11	12	13	15	16	19	21	23	27	18	28

☐ D

Opgave/Problem 70 (4%)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
11	5	19	13	28	22	15	29	30	17	2	18	12	14	10

Angiv resultatet af at anvende $\text{PARTITION}(A, 3, 14)$ på ovenstående array A .

Indicate the result of applying $\text{PARTITION}(A, 3, 14)$ to the above array A .

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
11	5	2	12	13	14	15	17	18	19	22	28	29	30	10

☐ A

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	5	10	11	12	13	14	15	17	18	19	22	28	29	30

☐ B

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
11	5	13	2	12	14	19	28	22	15	29	30	17	18	10

☐ C

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
11	5	13	2	12	14	15	29	30	17	19	18	28	22	10

☒ D

Radix-sort

Opgave/Problem 71 (4%)

2143 2102 0424 3324 1143 1224

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

2102 0424 3324 1224 2143 1143 ☒

2102 2143 1143 0424 3324 1224 ☐ B

0424 1143 1224 2102 2143 3324 ☐ C

0424 1143 1224 2143 2102 3324 ☐ D

2102 0424 1224 3324 1143 2143 ☐ E

Opgave/Problem 72 (4 %)

4010 3330 4410 1013 1430 1010

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 1010 | 1013 | 1430 | 3330 | 4010 | 4410 | ☐ A |
| 4010 | 4410 | 1010 | 1013 | 3330 | 1430 | ☒ B |
| 4010 | 4410 | 1010 | 3330 | 1430 | 1013 | ☐ C |
| 1013 | 1010 | 1430 | 3330 | 4010 | 4410 | ☐ D |
| 1010 | 4010 | 4410 | 1013 | 1430 | 3330 | ☐ E |

Opgave/Problem 73 (4 %)

2124 3404 2024 4324 1013 2013

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 1013 | 2013 | 3404 | 2124 | 2024 | 4324 | ☐ A |
| 1013 | 2024 | 2013 | 2124 | 3404 | 4324 | ☐ B |
| 1013 | 2013 | 2024 | 2124 | 3404 | 4324 | ☐ C |
| 3404 | 1013 | 2013 | 2024 | 2124 | 4324 | ☐ D |
| 3404 | 1013 | 2013 | 2124 | 2024 | 4324 | ☒ E |

Opgave/Problem 74 (4 %)

1143 4432 0234 1134 1432 4034

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|-------------------------------------|
| 4432 | 1432 | 0234 | 1134 | 4034 | 1143 | ☒ |
| 0234 | 1143 | 1134 | 1432 | 4034 | 4432 | ☐ |
| 0234 | 1134 | 1143 | 1432 | 4034 | 4432 | ☐ |
| 1432 | 4432 | 0234 | 1134 | 4034 | 1143 | ☐ |
| 4432 | 1432 | 1143 | 0234 | 1134 | 4034 | ☐ |

Opgave/Problem 75 (4 %)

2042 0041 4041 2241 4112 2212

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|-------------------------------------|
| 4112 | 2212 | 0041 | 4041 | 2241 | 2042 | ☒ |
| 0041 | 2042 | 2241 | 2212 | 4041 | 4112 | ☐ |
| 0041 | 2042 | 2212 | 2241 | 4041 | 4112 | ☐ |
| 0041 | 4041 | 2241 | 4112 | 2212 | 2042 | ☐ |
| 2212 | 4112 | 0041 | 2241 | 4041 | 2042 | ☐ |

Opgave/Problem 76 (4 %)

4040 2123 0430 0330 0423 4123

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 0430 | 0330 | 4040 | 2123 | 0423 | 4123 | ☐ A |
| 0330 | 0423 | 0430 | 2123 | 4040 | 4123 | ☐ B |
| 0330 | 0430 | 0423 | 2123 | 4040 | 4123 | ☐ C |
| 0423 | 2123 | 4123 | 0330 | 0430 | 4040 | ☐ D |
| 2123 | 0423 | 4123 | 0430 | 0330 | 4040 | ☒ E |

Opgave/Problem 77 (4 %)

3331 4143 2031 0031 4012 2012

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 4012 | 2012 | 3331 | 2031 | 0031 | 4143 | ☒ A |
| 0031 | 2031 | 2012 | 3331 | 4012 | 4143 | ☐ B |
| 0031 | 2012 | 2031 | 3331 | 4012 | 4143 | ☐ C |
| 2012 | 4012 | 0031 | 2031 | 3331 | 4143 | ☐ D |
| 3331 | 2031 | 0031 | 4012 | 2012 | 4143 | ☐ E |

Opgave/Problem 78 (4 %)

0321 4000 2121 2302 2300 1102

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 4000 | 2300 | 0321 | 2121 | 2302 | 1102 | ☐ A |
| 4000 | 2300 | 2302 | 1102 | 0321 | 2121 | ☒ B |
| 2300 | 4000 | 1102 | 2302 | 0321 | 2121 | ☐ C |
| 0321 | 1102 | 2121 | 2300 | 2302 | 4000 | ☐ D |
| 0321 | 1102 | 2121 | 2302 | 2300 | 4000 | ☐ E |

Opgave/Problem 79 (4 %)

4033 4313 2433 3120 4333 4320

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4$, $k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4$, $k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

- | | | | | | | |
|------|------|------|------|------|------|---------------------------------------|
| 4313 | 3120 | 4320 | 4033 | 2433 | 4333 | ☒ A |
| 4313 | 3120 | 4320 | 2433 | 4033 | 4333 | ☐ B |
| 2433 | 3120 | 4033 | 4313 | 4333 | 4320 | ☐ C |
| 3120 | 4320 | 4313 | 4033 | 2433 | 4333 | ☐ D |
| 2433 | 3120 | 4033 | 4313 | 4320 | 4333 | ☐ E |

Opgave/*Problem* 80 (4%)

3241 2123 1023 0023 3233 0433

Betragt RADIX-SORT anvendt på ovenstående liste af tal ($d = 4, k = 5$). Angiv den delvist sorterede liste efter at RADIX-SORT har sorteret tallene efter de *to* mindstbetydende cifre.

Consider RADIX-SORT applied to the above list of numbers ($d = 4, k = 5$). Indicate the partially sorted list after RADIX-SORT has sorted the numbers according to the two least significant digits.

0023	0433	1023	2123	3241	3233	☐ A
2123	1023	0023	3233	0433	3241	☒
3241	2123	1023	0023	3233	0433	☐ C
0023	0433	1023	2123	3233	3241	☐ D
0023	1023	2123	0433	3233	3241	☐ E

Lineær probing/*Linear probing*

Opgave/*Problem* 81 (4%)

0	1	2	3	4	5	6	7	8	9	10
11				5	16				3	14

I ovenstående hashtabel af størrelse 11 er anvendt *lineær probing* med hashfunktionen $h(k) = 3k \bmod 11$.

Angiv positionerne de fem elementer 0, 2, 7, 8 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 3, 5, 11, 14 og 16).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 3k \bmod 11$. Indicate the positions the five elements 0, 2, 7, 8 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 3, 5, 11, 14 and 16).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	81.1
INSERT(2)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒	☐ H	☐ I	☐ J	☐ K	81.2
INSERT(7)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	81.3
INSERT(8)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	81.4
INSERT(10)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒	☐ J	☐ K	81.5

Opgave/Problem 82 (4%)

0	1	2	3	4	5	6	7	8	9	10
22	6							15	4	21

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

Angiv positionerne de fem elementer 1, 2, 3, 5 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 4, 6, 15, 21 og 22).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 2k \bmod 11$. Indicate the positions the five elements 1, 2, 3, 5 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 4, 6, 15, 21 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(1)	A	B	⊗	D	E	F	G	H	I	J	K	82.1
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	82.2
INSERT(3)	A	B	C	D	E	F	⊗	H	I	J	K	82.3
INSERT(5)	A	B	⊗	D	E	F	G	H	I	J	K	82.4
INSERT(11)	A	B	⊗	D	E	F	G	H	I	J	K	82.5

Opgave/Problem 83 (4%)

0	1	2	3	4	5	6	7	8	9	10
22	0							6	17	15

I ovenstående hashtabel af størrelse 11 er anvendt *linear probing* med hashfunktionen $h(k) = 5k \bmod 11$.

Angiv positionerne de fem elementer 1, 3, 4, 9 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 6, 15, 17 og 22).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 5k \bmod 11$. Indicate the positions the five elements 1, 3, 4, 9 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 6, 15, 17 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(1)	A	B	C	D	E	⊗	G	H	I	J	K	83.1
INSERT(3)	A	B	C	D	⊗	F	G	H	I	J	K	83.2
INSERT(4)	A	B	⊗	D	E	F	G	H	I	J	K	83.3
INSERT(9)	A	B	⊗	D	E	F	G	H	I	J	K	83.4
INSERT(10)	A	B	C	D	E	F	⊗	H	I	J	K	83.5

Opgave/Problem 84 (4%)

0	1	2	3	4	5	6	7	8	9	10
11	22	20							4	13

I ovenstående hashtabel af størrelse 11 er anvendt *linear probing* med hashfunktionen $h(k) = 5k \bmod 11$.

Angiv positionerne de fem elementer 0, 2, 3, 5 og 6 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 4, 11, 13, 20 og 22).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 5k \bmod 11$. Indicate the positions the five elements 0, 2, 3, 5 and 6 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 4, 11, 13, 20 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	B	C	⊗	E	F	G	H	I	J	K	84.1
INSERT(2)	A	B	C	⊗	E	F	G	H	I	J	K	84.2
INSERT(3)	A	B	C	D	⊗	F	G	H	I	J	K	84.3
INSERT(5)	A	B	C	⊗	E	F	G	H	I	J	K	84.4
INSERT(6)	A	B	C	D	E	F	G	H	⊗	J	K	84.5

Opgave/Problem 85 (4%)

0	1	2	3	4	5	6	7	8	9	10
		1				3	14		21	5

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

Angiv positionerne de fem elementer 2, 4, 6, 9 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 3, 5, 14 og 21).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 2k \bmod 11$. Indicate the positions the five elements 2, 4, 6, 9 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 3, 5, 14 and 21).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	85.1
INSERT(4)	A	B	C	D	E	F	G	H	⊗	J	K	85.2
INSERT(6)	A	⊗	C	D	E	F	G	H	I	J	K	85.3
INSERT(9)	A	B	C	D	E	F	G	H	⊗	J	K	85.4
INSERT(10)	⊗	B	C	D	E	F	G	H	I	J	K	85.5

Opgave/Problem 86 (4%)

0	1	2	3	4	5	6	7	8	9	10
11	0					21			4	15

I ovenstående hashtabel af størrelse 11 er anvendt *lineær probing* med hashfunktionen $h(k) = 5k \bmod 11$.

Angiv positionerne de fem elementer 2, 3, 5, 7 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 4, 11, 15 og 21).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 5k \bmod 11$. Indicate the positions the five elements 2, 3, 5, 7 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 4, 11, 15 and 21).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	⊗	D	E	F	G	H	I	J	K	86.1
INSERT(3)	A	B	C	D	⊗	F	G	H	I	J	K	86.2
INSERT(5)	A	B	C	⊗	E	F	G	H	I	J	K	86.3
INSERT(7)	A	B	⊗	D	E	F	G	H	I	J	K	86.4
INSERT(9)	A	B	⊗	D	E	F	G	H	I	J	K	86.5

Opgave/Problem 87 (4%)

0	1	2	3	4	5	6	7	8	9	10
0				2		14	3			16

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

Angiv positionerne de fem elementer 4, 5, 6, 10 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 2, 3, 14 og 16).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 2k \bmod 11$. Indicate the positions the five elements 4, 5, 6, 10 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 2, 3, 14 and 16).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(4)	A	B	C	D	E	F	G	H	⊗	J	K	87.1
INSERT(5)	A	⊗	C	D	E	F	G	H	I	J	K	87.2
INSERT(6)	A	⊗	C	D	E	F	G	H	I	J	K	87.3
INSERT(10)	A	B	C	D	E	F	G	H	I	⊗	K	87.4
INSERT(11)	A	⊗	C	D	E	F	G	H	I	J	K	87.5

Opgave/Problem 88 (4 %)

0	1	2	3	4	5	6	7	8	9	10
22			7	18				15		16

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

Angiv positionerne de fem elementer 0, 2, 3, 5 og 6 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 7, 15, 16, 18 og 22).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 2k \bmod 11$. Indicate the positions the five elements 0, 2, 3, 5 and 6 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 7, 15, 16, 18 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	✗	C	D	E	F	G	H	I	J	K	88.1
INSERT(2)	A	B	C	D	E	✗	G	H	I	J	K	88.2
INSERT(3)	A	B	C	D	E	F	✗	H	I	J	K	88.3
INSERT(5)	A	✗	C	D	E	F	G	H	I	J	K	88.4
INSERT(6)	A	✗	C	D	E	F	G	H	I	J	K	88.5

Opgave/Problem 89 (4 %)

0	1	2	3	4	5	6	7	8	9	10
22	20	0							15	2

I ovenstående hashtabel af størrelse 11 er anvendt *lineær probing* med hashfunktionen $h(k) = 5k \bmod 11$.

Angiv positionerne de fem elementer 4, 6, 7, 8 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 2, 15, 20 og 22).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 5k \bmod 11$. Indicate the positions the five elements 4, 6, 7, 8 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 2, 15, 20 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(4)	A	B	C	✗	E	F	G	H	I	J	K	89.1
INSERT(6)	A	B	C	D	E	F	G	H	✗	J	K	89.2
INSERT(7)	A	B	C	✗	E	F	G	H	I	J	K	89.3
INSERT(8)	A	B	C	D	E	F	G	✗	I	J	K	89.4
INSERT(9)	A	B	C	✗	E	F	G	H	I	J	K	89.5

Opgave/Problem 90 (4 %)

0	1	2	3	4	5	6	7	8	9	10
		1			19			15	4	16

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

Angiv positionerne de fem elementer 2, 6, 7, 8 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 4, 15, 16 og 19).

In the above hash table of size 11, linear probing is used with hash function $h(k) = 2k \bmod 11$. Indicate the positions the five elements 2, 6, 7, 8 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 4, 15, 16 and 19).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	90.1
INSERT(6)	A	⊗	C	D	E	F	G	H	I	J	K	90.2
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	90.3
INSERT(8)	A	B	C	D	E	F	⊗	H	I	J	K	90.4
INSERT(10)	⊗	B	C	D	E	F	G	H	I	J	K	90.5

Kvadratisk probing/ *Quadratic probing*

Opgave/ *Problem 91* (4 %)

0	1	2	3	4	5	6	7	8	9	10
		1	12			13			18	7

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 3k \bmod 11$ og $h(k, i) = (h'(k) + 5i + 5i^2) \bmod 11$.

Angiv positionerne de fem elementer 0, 2, 3, 5 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 7, 12, 13 og 18).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 3k \bmod 11$ and $h(k, i) = (h'(k) + 5i + 5i^2) \bmod 11$. Indicate the positions the five elements 0, 2, 3, 5 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 7, 12, 13 and 18).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	⊗	B	C	D	E	F	G	H	I	J	K	91.1
INSERT(2)	A	B	C	D	E	⊗	G	H	I	J	K	91.2
INSERT(3)	A	B	C	D	E	F	G	H	⊗	J	K	91.3
INSERT(5)	A	B	C	D	⊗	F	G	H	I	J	K	91.4
INSERT(9)	A	B	C	D	E	⊗	G	H	I	J	K	91.5

Opgave/ *Problem 92* (4 %)

0	1	2	3	4	5	6	7	8	9	10
22					0	18	10		5	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 4k \bmod 11$ og $h(k, i) = (h'(k) + 2i + 3i^2) \bmod 11$.

Angiv positionerne de fem elementer 2, 3, 7, 9 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 5, 10, 18 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 4k \bmod 11$ and $h(k, i) = (h'(k) + 2i + 3i^2) \bmod 11$. Indicate the positions the five elements 2, 3, 7, 9 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 5, 10, 18 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	E	F	G	H	⊗	J	K	92.1
INSERT(3)	A	⊗	C	D	E	F	G	H	I	J	K	92.2
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	92.3
INSERT(9)	A	B	C	⊗	E	F	G	H	I	J	K	92.4
INSERT(11)	A	⊗	C	D	E	F	G	H	I	J	K	92.5

Opgave/Problem 93 (4%)

0	1	2	3	4	5	6	7	8	9	10
22	6					10		15	17	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 2k \bmod 11$ og $h(k, i) = (h'(k) + 3i + 5i^2) \bmod 11$.

Angiv positionerne de fem elementer 0, 2, 4, 7 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 6, 10, 15, 17 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 2k \bmod 11$ and $h(k, i) = (h'(k) + 3i + 5i^2) \bmod 11$. Indicate the positions the five elements 0, 2, 4, 7 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 6, 10, 15, 17 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	B	C	D	⊗	F	G	H	I	J	K	93.1
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	93.2
INSERT(4)	A	B	C	D	E	⊗	G	H	I	J	K	93.3
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	93.4
INSERT(9)	A	B	C	D	E	F	G	⊗	I	J	K	93.5

Opgave/Problem 94 (4%)

0	1	2	3	4	5	6	7	8	9	10
	15	6			20		17			18

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 3k \bmod 11$ og $h(k, i) = (h'(k) + 2i + 4i^2) \bmod 11$.

Angiv positionerne de fem elementer 0, 4, 7, 9 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 6, 15, 17, 18 og 20).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 3k \bmod 11$ and $h(k, i) = (h'(k) + 2i + 4i^2) \bmod 11$. Indicate the positions the five elements 0, 4, 7, 9 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 6, 15, 17, 18 and 20).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	⊗	B	C	D	E	F	G	H	I	J	K	94.1
INSERT(4)	A	B	C	⊗	E	F	G	H	I	J	K	94.2
INSERT(7)	A	B	C	D	E	F	G	H	⊗	J	K	94.3
INSERT(9)	⊗	B	C	D	E	F	G	H	I	J	K	94.4
INSERT(10)	A	B	C	D	E	F	G	H	⊗	J	K	94.5

Opgave/Problem 95 (4%)

0	1	2	3	4	5	6	7	8	9	10
0					12	10		17	11	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 5k \bmod 11$ og $h(k, i) = (h'(k) + 5i^2) \bmod 11$.

Angiv positionerne de fem elementer 1, 3, 4, 5 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 10, 11, 12 og 17).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 5k \bmod 11$ and $h(k, i) = (h'(k) + 5i^2) \bmod 11$. Indicate the positions the five elements 1, 3, 4, 5 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 10, 11, 12 and 17).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(1)	A	B	C	D	E	F	G	H	I	J	⊗	95.1
INSERT(3)	A	B	C	D	⊗	F	G	H	I	J	K	95.2
INSERT(4)	A	B	C	⊗	E	F	G	H	I	J	K	95.3
INSERT(5)	A	B	C	⊗	E	F	G	H	I	J	K	95.4
INSERT(9)	A	⊗	C	D	E	F	G	H	I	J	K	95.5

Opgave/Problem 96 (4%)

0	1	2	3	4	5	6	7	8	9	10
22		11				3		4	21	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 2k \bmod 11$ og $h(k, i) = (h'(k) + i + i^2) \bmod 11$.

Angiv positionerne de fem elementer 1, 2, 5, 7 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 3, 4, 11, 21 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 2k \bmod 11$ and $h(k, i) = (h'(k) + i + i^2) \bmod 11$. Indicate the positions the five elements 1, 2, 5, 7 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 3, 4, 11, 21 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(1)	A	B	C	D	⊗	F	G	H	I	J	K	96.1
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	96.2
INSERT(5)	A	B	C	D	E	F	G	H	I	J	⊗	96.3
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	96.4
INSERT(10)	A	B	C	D	⊗	F	G	H	I	J	K	96.5

Opgave/Problem 97 (4%)

0	1	2	3	4	5	6	7	8	9	10
3	17					14	20		21	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 2k \bmod 11$ og $h(k, i) = (h'(k) + 3i + 2i^2) \bmod 11$.

Angiv positionerne de fem elementer 2, 4, 7, 9 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 3, 14, 17, 20 og 21).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 2k \bmod 11$ and $h(k, i) = (h'(k) + 3i + 2i^2) \bmod 11$. Indicate the positions the five elements 2, 4, 7, 9 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 3, 14, 17, 20 and 21).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	97.1
INSERT(4)	A	B	C	D	E	F	G	H	⊗	J	K	97.2
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	97.3
INSERT(9)	A	B	C	D	E	F	G	H	I	J	⊗	97.4
INSERT(10)	A	B	C	⊗	E	F	G	H	I	J	K	97.5

Opgave/Problem 98 (4%)

0	1	2	3	4	5	6	7	8	9	10
22		18					19	6	11	

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 5k \bmod 11$ og $h(k, i) = (h'(k) + 5i + 2i^2) \bmod 11$.

Angiv positionerne de fem elementer 0, 1, 2, 3 og 8 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 6, 11, 18, 19 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 5k \bmod 11$ and $h(k, i) = (h'(k) + 5i + 2i^2) \bmod 11$. Indicate the positions the five elements 0, 1, 2, 3 and 8 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 6, 11, 18, 19 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	B	C	⊗	E	F	G	H	I	J	K	98.1
INSERT(1)	A	B	C	D	E	⊗	G	H	I	J	K	98.2
INSERT(2)	A	B	C	D	E	F	G	H	I	J	⊗	98.3
INSERT(3)	A	B	C	⊗	F	G	H	I	J	K		98.4
INSERT(8)	A	B	C	⊗	E	F	G	H	I	J	K	98.5

Opgave/Problem 99 (4 %)

0	1	2	3	4	5	6	7	8	9	10
11			18	13			20	22		

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 2k \bmod 11$ og $h(k, i) = (h'(k) + 3i + 5i^2) \bmod 11$.

Angiv positionerne de fem elementer 2, 5, 6, 7 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 11, 13, 18, 20 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 2k \bmod 11$ and $h(k, i) = (h'(k) + 3i + 5i^2) \bmod 11$. Indicate the positions the five elements 2, 5, 6, 7 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 11, 13, 18, 20 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	99.1
INSERT(5)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☒	99.2
INSERT(6)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	99.3
INSERT(7)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	99.4
INSERT(9)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒	☐ H	☐ I	☐ J	☐ K	99.5

Opgave/Problem 100 (4 %)

0	1	2	3	4	5	6	7	8	9	10
0	22			13	8			15		

I ovenstående hashtabel af størrelse 11 er anvendt *kvadratisk probing* med hashfunktionerne $h'(k) = 2k \bmod 11$ og $h(k, i) = (h'(k) + i^2) \bmod 11$.

Angiv positionerne de fem elementer 2, 4, 6, 9 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 8, 13, 15 og 22).

In the above hash table of size 11, quadratic probing is used with hash functions $h'(k) = 2k \bmod 11$ and $h(k, i) = (h'(k) + i^2) \bmod 11$. Indicate the positions the five elements 2, 4, 6, 9 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 8, 13, 15 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	100.1
INSERT(4)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☒	☐ K	100.2
INSERT(6)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	100.3
INSERT(9)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	☐ I	☐ J	☐ K	100.4
INSERT(11)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☒	☐ K	100.5

Dobbelt hashing/*Double hashing*

Opgave/*Problem* 101 (4 %)

0	1	2	3	4	5	6	7	8	9	10
		19	12		14		6		3	

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 3k \bmod 11$ og $h_2(k) = 1 + (4k \bmod 10)$.

Angiv positionerne de fem elementer 2, 5, 8, 9 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 3, 6, 12, 14 og 19).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 3k \bmod 11$ and $h_2(k) = 1 + (4k \bmod 10)$. Indicate the positions the five elements 2, 5, 8, 9 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 3, 6, 12, 14 and 19).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	E	F	⊗	H	I	J	K	101.1
INSERT(5)	A	B	C	D	⊗	F	G	H	I	J	K	101.2
INSERT(8)	A	B	C	D	E	F	G	H	⊗	J	K	101.3
INSERT(9)	A	⊗	C	D	E	F	G	H	I	J	K	101.4
INSERT(11)	⊗	B	C	D	E	F	G	H	I	J	K	101.5

Opgave/*Problem* 102 (4 %)

0	1	2	3	4	5	6	7	8	9	10
		19	7		8		9	18		

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 2k \bmod 11$ og $h_2(k) = 1 + (3k \bmod 10)$.

Angiv positionerne de fem elementer 1, 2, 4, 6 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 7, 8, 9, 18 og 19).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 2k \bmod 11$ and $h_2(k) = 1 + (3k \bmod 10)$. Indicate the positions the five elements 1, 2, 4, 6 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 7, 8, 9, 18 and 19).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(1)	A	B	C	D	E	F	⊗	H	I	J	K	102.1
INSERT(2)	A	B	C	D	⊗	F	G	H	I	J	K	102.2
INSERT(4)	⊗	B	C	D	E	F	G	H	I	J	K	102.3
INSERT(6)	A	⊗	C	D	E	F	G	H	I	J	K	102.4
INSERT(11)	⊗	B	C	D	E	F	G	H	I	J	K	102.5

Opgave/Problem 103 (4 %)

0	1	2	3	4	5	6	7	8	9	10
11	9			13			22			16

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 2k \bmod 11$ og $h_2(k) = 1 + (3k \bmod 10)$.

Angiv positionerne de fem elementer 0, 4, 5, 7 og 8 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 9, 11, 13, 16 og 22).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 2k \bmod 11$ and $h_2(k) = 1 + (3k \bmod 10)$. Indicate the positions the five elements 0, 4, 5, 7 and 8 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 9, 11, 13, 16 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	B	⊗	D	E	F	G	H	I	J	K	103.1
INSERT(4)	A	B	C	D	E	F	G	H	⊗	J	K	103.2
INSERT(5)	A	B	C	D	E	⊗	G	H	I	J	K	103.3
INSERT(7)	A	B	C	⊗	E	F	G	H	I	J	K	103.4
INSERT(8)	A	B	C	D	E	⊗	G	H	I	J	K	103.5

Opgave/Problem 104 (4 %)

0	1	2	3	4	5	6	7	8	9	10
10							21	19	16	8

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 4k \bmod 11$ og $h_2(k) = 1 + (2k \bmod 10)$.

Angiv positionerne de fem elementer 0, 1, 2, 3 og 4 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 8, 10, 16, 19 og 21).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 4k \bmod 11$ and $h_2(k) = 1 + (2k \bmod 10)$. Indicate the positions the five elements 0, 1, 2, 3 and 4 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 8, 10, 16, 19 and 21).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	⊗	C	D	E	F	G	H	I	J	K	104.1
INSERT(1)	A	B	C	D	⊗	F	G	H	I	J	K	104.2
INSERT(2)	A	B	⊗	D	E	F	G	H	I	J	K	104.3
INSERT(3)	A	⊗	C	D	E	F	G	H	I	J	K	104.4
INSERT(4)	A	B	C	D	E	⊗	G	H	I	J	K	104.5

Opgave/Problem 105 (4 %)

0	1	2	3	4	5	6	7	8	9	10
22	0	6							5	19

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 4k \bmod 11$ og $h_2(k) = 1 + (3k \bmod 10)$.

Angiv positionerne de fem elementer 2, 3, 4, 8 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 5, 6, 19 og 22).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 4k \bmod 11$ and $h_2(k) = 1 + (3k \bmod 10)$. Indicate the positions the five elements 2, 3, 4, 8 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 5, 6, 19 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(2)	A	B	C	D	E	F	G	H	⊗	J	K	105.1
INSERT(3)	A	B	C	D	E	F	G	H	⊗	J	K	105.2
INSERT(4)	A	B	C	D	E	⊗	G	H	I	J	K	105.3
INSERT(8)	A	B	C	D	⊗	F	G	H	I	J	K	105.4
INSERT(9)	A	B	C	⊗	E	F	G	H	I	J	K	105.5

Opgave/Problem 106 (4 %)

0	1	2	3	4	5	6	7	8	9	10
11				1	7	18		2		

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 4k \bmod 11$ og $h_2(k) = 1 + (2k \bmod 10)$.

Angiv positionerne de fem elementer 0, 3, 4, 5 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 2, 7, 11 og 18).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 4k \bmod 11$ and $h_2(k) = 1 + (2k \bmod 10)$. Indicate the positions the five elements 0, 3, 4, 5 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 2, 7, 11 and 18).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	A	⊗	C	D	E	F	G	H	I	J	K	106.1
INSERT(3)	A	⊗	C	D	E	F	G	H	I	J	K	106.2
INSERT(4)	A	B	C	⊗	E	F	G	H	I	J	K	106.3
INSERT(5)	A	B	C	D	E	F	G	H	I	⊗	K	106.4
INSERT(9)	A	B	C	⊗	E	F	G	H	I	J	K	106.5

Opgave/Problem 107 (4 %)

0	1	2	3	4	5	6	7	8	9	10
11	14			1		7				3

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 4k \bmod 11$ og $h_2(k) = 1 + (3k \bmod 10)$.

Angiv positionerne de fem elementer 0, 4, 6, 8 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 3, 7, 11 og 14).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 4k \bmod 11$ and $h_2(k) = 1 + (3k \bmod 10)$. Indicate the positions the five elements 0, 4, 6, 8 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 3, 7, 11 and 14).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	107.1
INSERT(4)	☐ A	☐ B	☐ C	☐ D	☐ E	☒	☐ G	☐ H	☐ I	☐ J	☐ K	107.2
INSERT(6)	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	107.3
INSERT(8)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☒	☐ K	107.4
INSERT(10)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	☐ I	☐ J	☐ K	107.5

Opgave/Problem 108 (4 %)

0	1	2	3	4	5	6	7	8	9	10
22			5		12		8			11

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 5k \bmod 11$ og $h_2(k) = 1 + (4k \bmod 10)$.

Angiv positionerne de fem elementer 0, 1, 2, 6 og 9 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 5, 8, 11, 12 og 22).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 5k \bmod 11$ and $h_2(k) = 1 + (4k \bmod 10)$. Indicate the positions the five elements 0, 1, 2, 6 and 9 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 5, 8, 11, 12 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(0)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	108.1
INSERT(1)	☐ A	☐ B	☐ C	☐ D	☒	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	108.2
INSERT(2)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒	☐ J	☐ K	108.3
INSERT(6)	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒	☐ J	☐ K	108.4
INSERT(9)	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	☐ J	☐ K	108.5

Opgave/Problem 109 (4 %)

0	1	2	3	4	5	6	7	8	9	10
22		1			12				21	10

I ovenstående hashtabel af størrelse 11 er anvendt *dobbelt hashing* med hashfunktionerne $h_1(k) = 2k \bmod 11$ og $h_2(k) = 1 + (3k \bmod 10)$.

Angiv positionerne de fem elementer 3, 4, 5, 6 og 11 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 1, 10, 12, 21 og 22).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 2k \bmod 11$ and $h_2(k) = 1 + (3k \bmod 10)$. Indicate the positions the five elements 3, 4, 5, 6 and 11 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 1, 10, 12, 21 and 22).

	0	1	2	3	4	5	6	7	8	9	10	
INSERT(3)	A	B	C	D	E	F	⊗	H	I	J	K	109.1
INSERT(4)	A	B	C	D	E	F	G	H	⊗	J	K	109.2
INSERT(5)	A	B	C	D	E	F	⊗	H	I	J	K	109.3
INSERT(6)	A	⊗	C	D	E	F	G	H	I	J	K	109.4
INSERT(11)	A	B	C	D	⊗	F	G	H	I	J	K	109.5

Opgave/Problem 110 (4 %)

0	1	2	3	4	5	6	7	8	9	10
0		1	18		22		12			

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 3, 7, 8, 9 og 10 vil blive indsat på i hashtabellen (for hver af indsættelserne antager vi at hashtabellen kun indeholder elementerne 0, 1, 12, 18 og 22).

In the above hash table of size 11, double hashing is used with hash functions $h_1(k) = 2k \bmod 11$ and $h_2(k) = 1 + (2k \bmod 10)$. Indicate the positions the five elements 3, 7, 8, 9 and 10 will be inserted at in the hash table (for each insertion, we assume the hash table only contains the elements 0, 1, 12, 18 and 22).

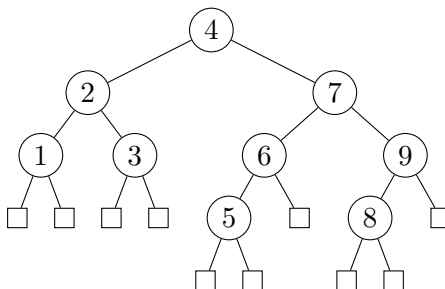
	0	1	2	3	4	5	6	7	8	9	10	
INSERT(3)	A	B	C	D	E	F	⊗	H	I	J	K	110.1
INSERT(7)	A	B	C	D	E	F	G	H	⊗	J	K	110.2
INSERT(8)	A	⊗	C	D	E	F	G	H	I	J	K	110.3
INSERT(9)	A	⊗	C	D	E	F	G	H	I	J	K	110.4
INSERT(10)	A	B	C	D	E	F	G	H	I	⊗	K	110.5

Lovlige rød-sortede træer/ *Valid red-black trees*

Opgave/Problem 111 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

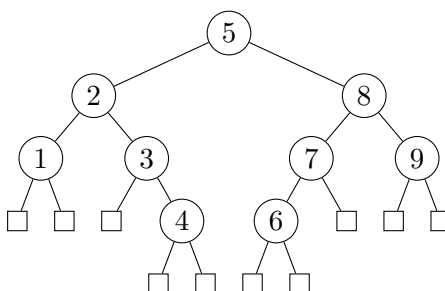


	Ja/Yes	Nej/No	
1, 3, 5, 7, 8	☒	☐	111.1
5, 8	☒	☐	111.2
4, 5, 8	☐	☒	111.3
2, 5, 6, 8, 9	☐	☒	111.4
2, 5, 7, 8	☒	☐	111.5

Opgave/Problem 112 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

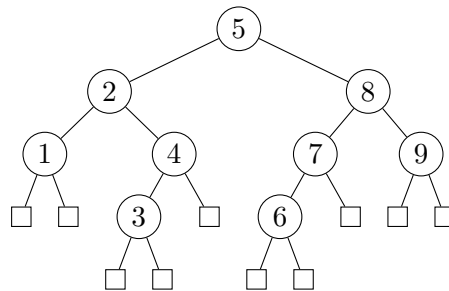


	Ja/Yes	Nej/No	
2, 4, 6, 7, 9	☐	☒	112.1
1, 3, 4, 6, 8	☐	☒	112.2
4, 5, 6	☐	☒	112.3
2, 4, 6, 8	☒	☐	112.4
4, 6	☒	☐	112.5

Opgave/Problem 113 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

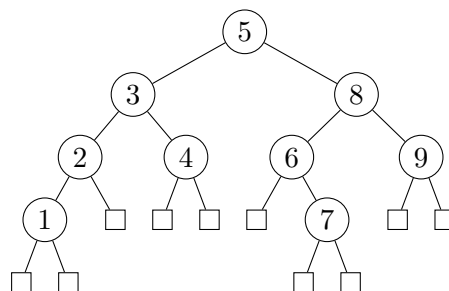


	Ja/Yes	Nej/No	
3, 5, 6	☐ A	☒ B	113.1
2, 3, 6, 7, 9	☐ A	☒ B	113.2
2, 3, 6, 8	☒ B	☐ A	113.3
1, 3, 4, 6, 8	☐ A	☒ B	113.4
3, 6	☒ B	☐ A	113.5

Opgave/Problem 114 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

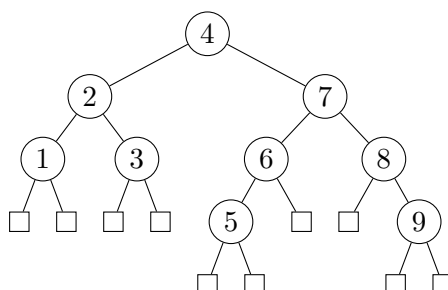


	Ja/Yes	Nej/No	
1, 7	☒ B	☐ A	114.1
1, 5, 7	☐ A	☒ B	114.2
1, 3, 7, 8	☒ B	☐ A	114.3
1, 2, 4, 7, 8	☐ A	☒ B	114.4
1, 3, 6, 7, 9	☐ A	☒ B	114.5

Opgave/Problem 115 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

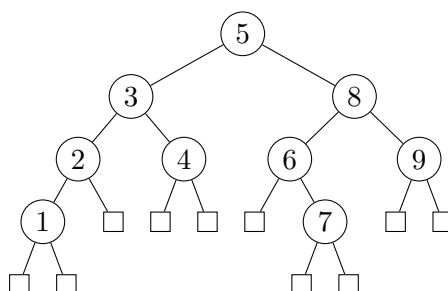


	Ja/Yes	Nej/No	
4, 5, 9	☐ A	☒ B	115.1
1, 3, 5, 6, 8, 9	☐ A	☒ B	115.2
1, 3, 5, 7, 9	☒ A	☐ B	115.3
5, 9	☒ A	☐ B	115.4
2, 5, 7, 9	☒ A	☐ B	115.5

Opgave/Problem 116 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

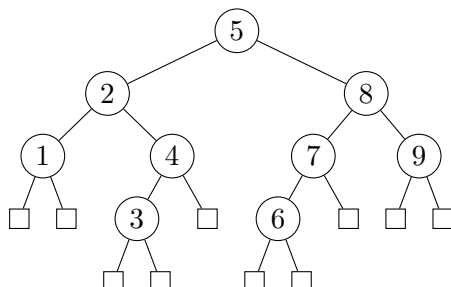


	Ja/Yes	Nej/No	
1, 5, 7	☐ A	☒ B	116.1
1, 3, 7, 8	☒ A	☐ B	116.2
1, 2, 4, 7, 8	☐ A	☒ B	116.3
1, 7	☒ A	☐ B	116.4
1, 3, 6, 7, 9	☐ A	☒ B	116.5

Opgave/Problem 117 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

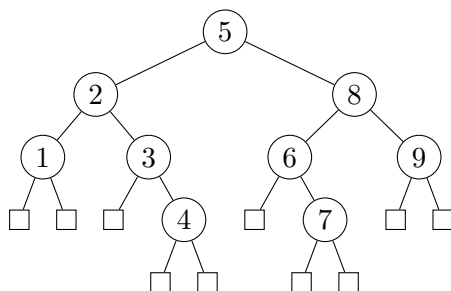


	Ja/Yes	Nej/No	
3, 6	☒	☐ B	117.1
1, 3, 4, 6, 8	☐ A	☒	117.2
3, 5, 6	☐ A	☒	117.3
2, 3, 6, 8	☒	☐ B	117.4
2, 3, 6, 7, 9	☐ A	☒	117.5

Opgave/Problem 118 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

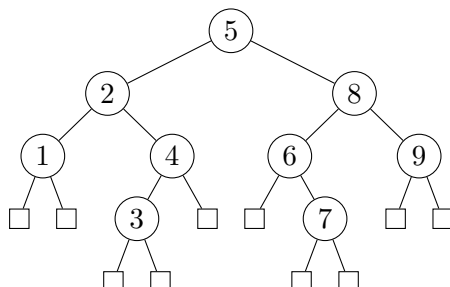


	Ja/Yes	Nej/No	
4, 7	☒	☐ B	118.1
1, 3, 4, 7, 8	☐ A	☒	118.2
2, 4, 7, 8	☒	☐ B	118.3
2, 4, 6, 7, 9	☐ A	☒	118.4
4, 5, 7	☐ A	☒	118.5

Opgave/Problem 119 (4 %)

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

For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.

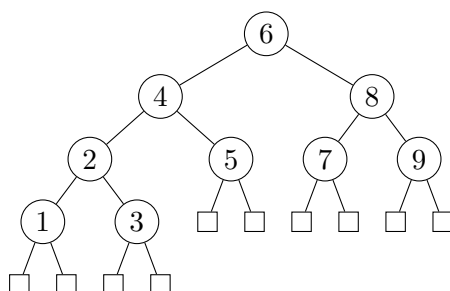


	Ja/Yes	Nej/No	
3, 7	☒	☐ B	119.1
1, 3, 4, 7, 8	☐ A	☒	119.2
3, 5, 7	☐ A	☒	119.3
2, 3, 7, 8	☒	☐ B	119.4
2, 3, 6, 7, 9	☐ A	☒	119.5

Opgave/Problem 120 (4 %)

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

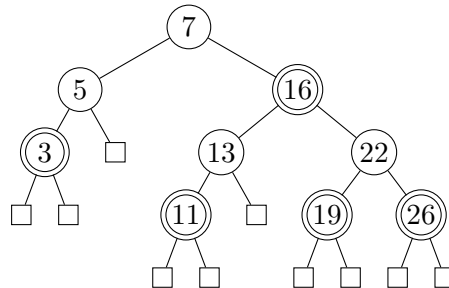
For each of the following subsets, indicate whether the binary tree below is a valid red-black tree if these nodes are colored red.



	Ja/Yes	Nej/No	
1, 3, 4, 7, 9	☒	☐ B	120.1
1, 3, 6	☐ A	☒	120.2
1, 2, 3, 5, 7, 9	☐ A	☒	120.3
1, 3, 4, 8	☒	☐ B	120.4
1, 3	☒	☐ B	120.5

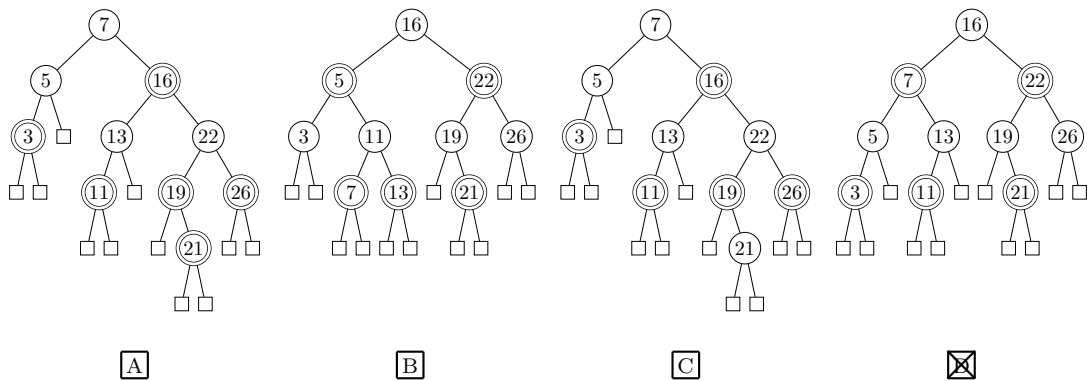
Rød-sort træ indsættelse/ *Red-black tree insertion*

Opgave/ *Problem 121* (4 %)

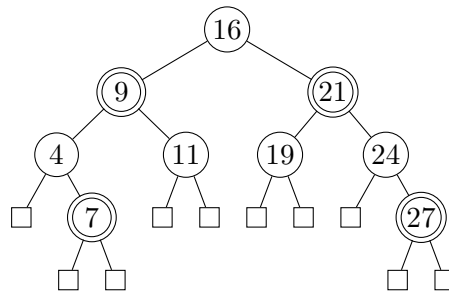


Angiv det resulterende rød-sortede træ når man indsætter 21 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 21 into the above red-black tree (double circles indicate red nodes).

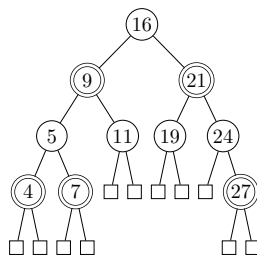


Opgave/Problem 122 (4 %)

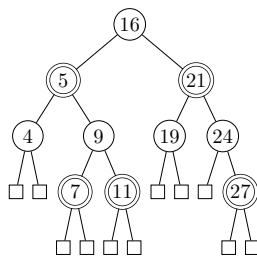


Angiv det resulterende rød-sort tr e n r man inds tter 5 i ovenst ende r d-sort tr e (dobbeltcirkler angiver r de knuder).

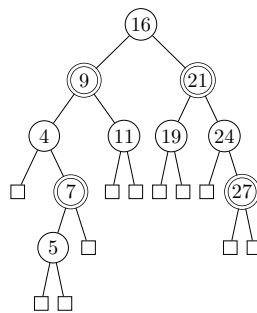
Indicate the resulting red-black tree when inserting 5 into the above red-black tree (double circles indicate red nodes).



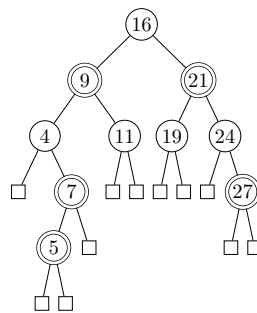
☒



☐

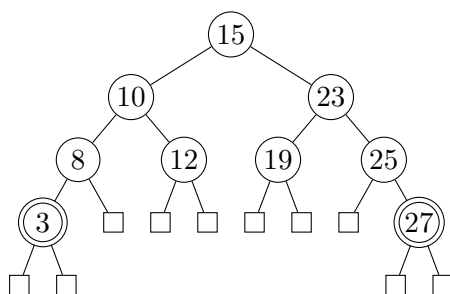


☐



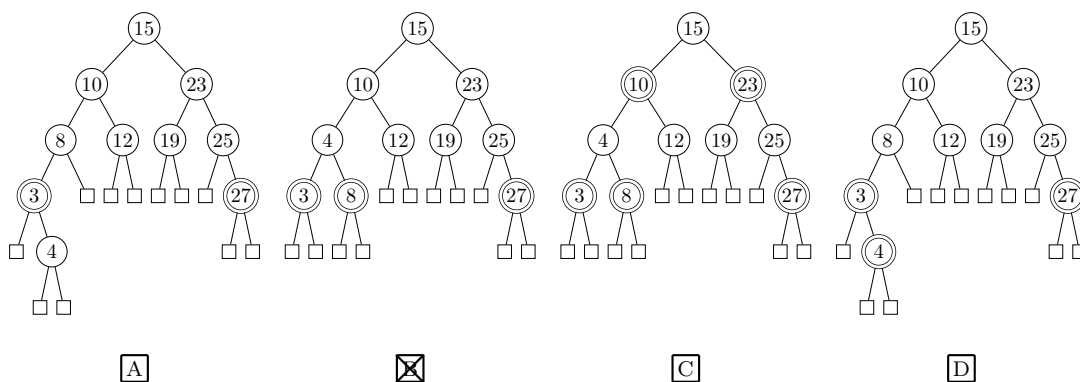
☐

Opgave/Problem 123 (4 %)

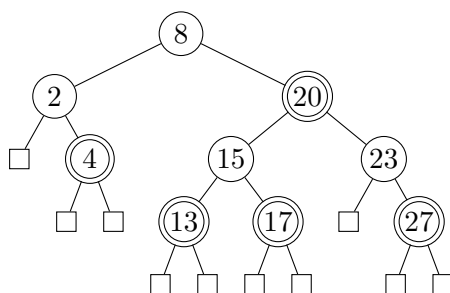


Angiv det resulterende rød-sortede træ når man indsætter 4 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 4 into the above red-black tree (double circles indicate red nodes).

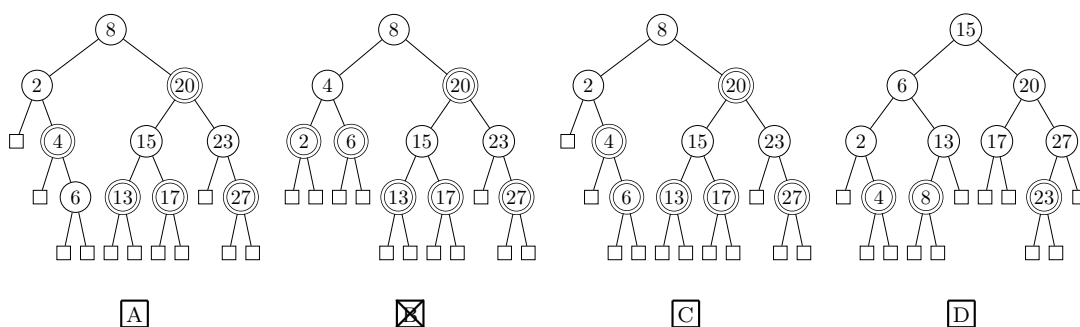


Opgave/Problem 124 (4 %)

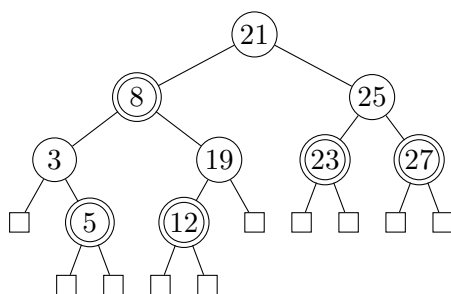


Angiv det resulterende rød-sortede træ når man indsætter 6 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 6 into the above red-black tree (double circles indicate red nodes).

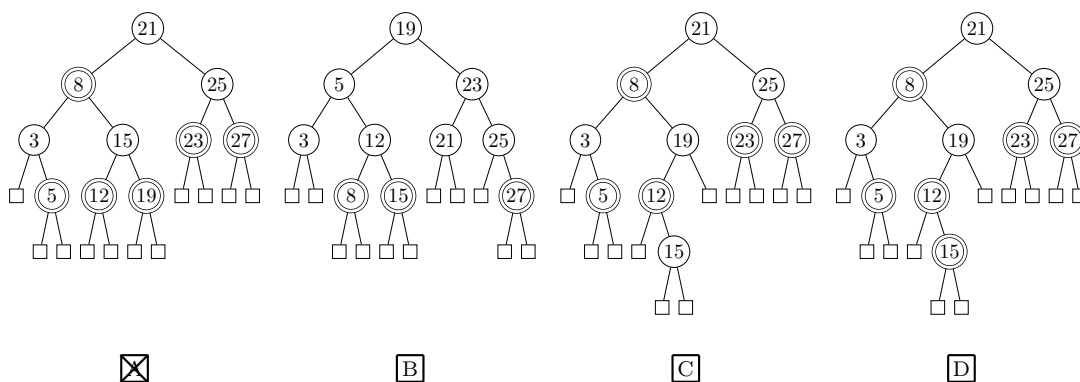


Opgave/Problem 125 (4 %)

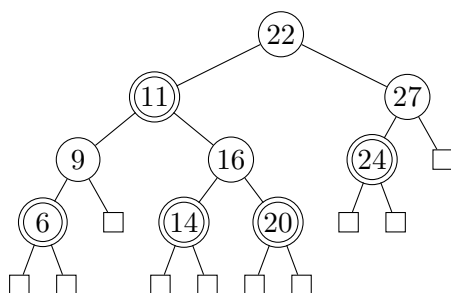


Angiv det resulterende rød-sortede træ når man indsætter 15 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 15 into the above red-black tree (double circles indicate red nodes).

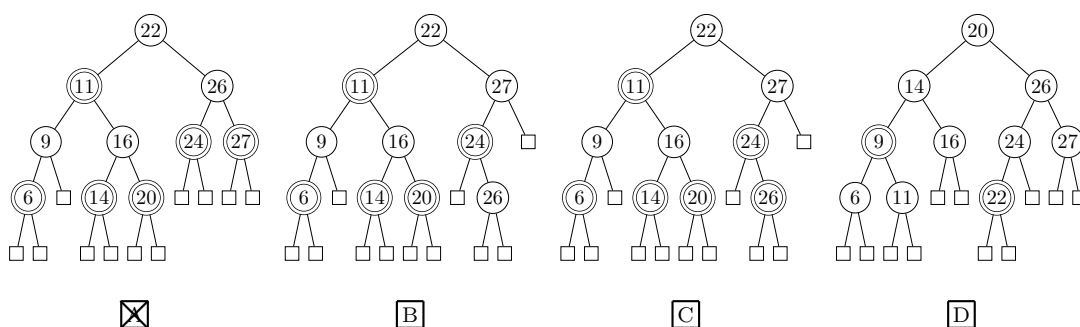


Opgave/Problem 126 (4 %)

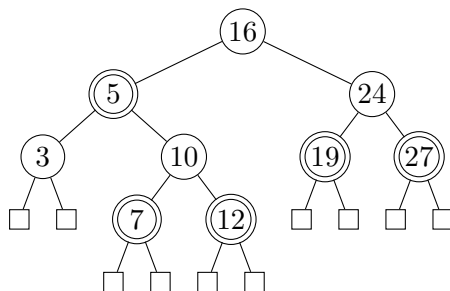


Angiv det resulterende rød-sortede træ når man indsætter 26 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 26 into the above red-black tree (double circles indicate red nodes).

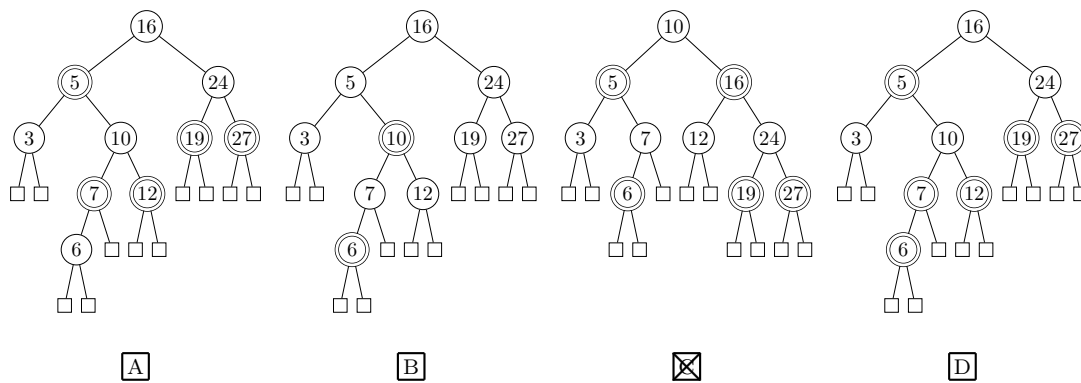


Opgave/Problem 127 (4 %)

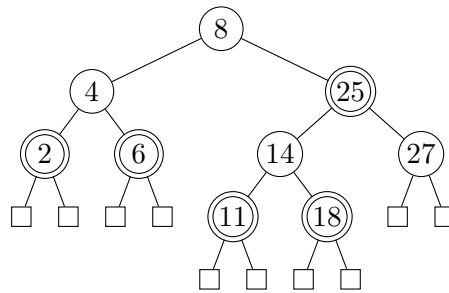


Angiv det resulterende rød-sortre træ når man indsætter 6 i ovenstående rød-sortre træ (dobbeltcirkler angiver røde knuder).

Indicate the resulting red-black tree when inserting 6 into the above red-black tree (double circles indicate red nodes).

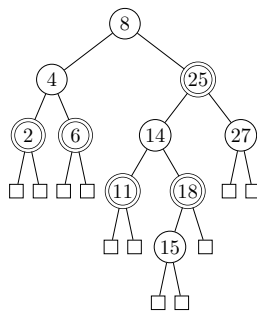


Opgave/Problem 128 (4 %)

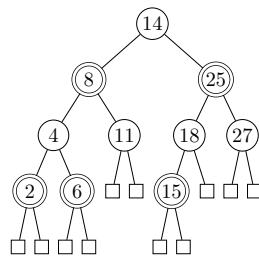


Angiv det resulterende rød-sortte træ når man indsætter 15 i ovenstående rød-sortte træ (dobbeltcirkler angiver røde knuder).

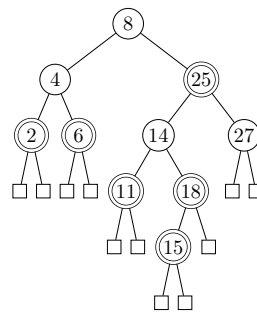
Indicate the resulting red-black tree when inserting 15 into the above red-black tree (double circles indicate red nodes).



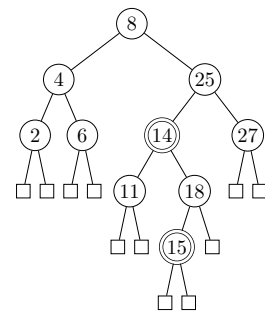
A



B

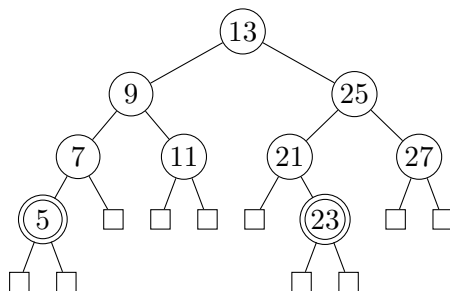


C



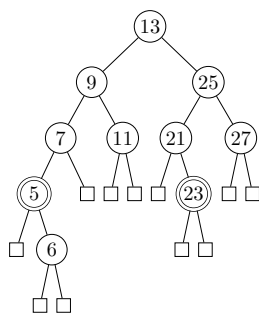
D

Opgave/Problem 129 (4 %)

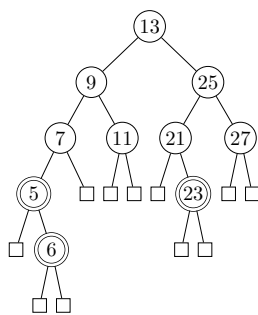


Angiv det resulterende rød-sortre træ når man indsætter 6 i ovenstående rød-sortre træ (dobbeltcirkler angiver røde knuder).

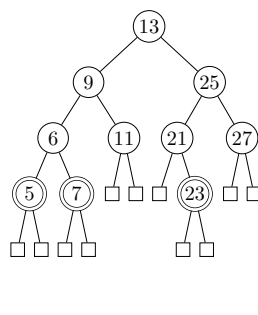
Indicate the resulting red-black tree when inserting 6 into the above red-black tree (double circles indicate red nodes).



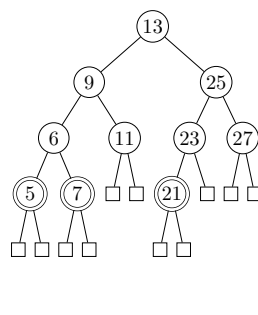
A



B

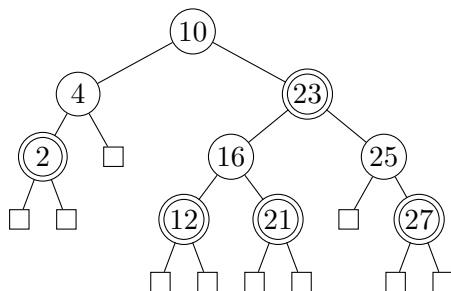


C



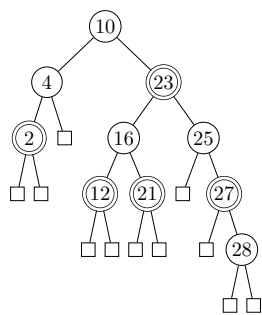
D

Opgave/Problem 130 (4 %)

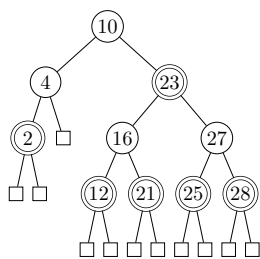


Angiv det resulterende rød-sortede træ når man indsætter 28 i ovenstående rød-sortede træ (dobbeltcirkler angiver røde knuder).

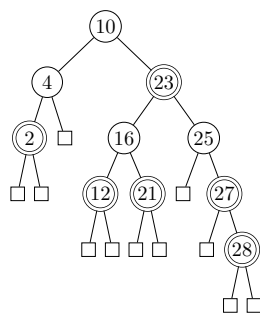
Indicate the resulting red-black tree when inserting 28 into the above red-black tree (double circles indicate red nodes).



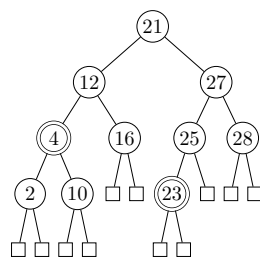
A



B



C



D

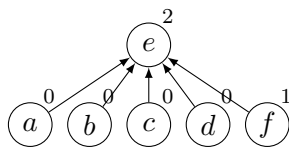
Union-find

Opgave/*Problem* 131 (4 %)

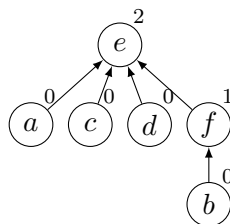
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

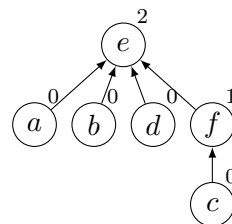
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(c, f)
 UNION(c, b)
 UNION(d, e)
 UNION(c, e)
 UNION(a, c)
 FIND-SET(b)



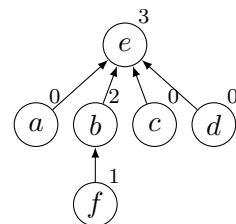
☒



☐



☐



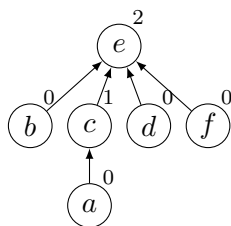
☐

Opgave/Problem 132 (4 %)

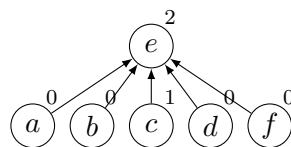
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

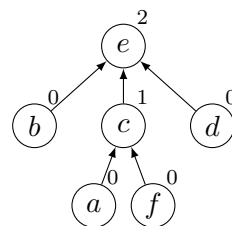
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(f, c)
 UNION(c, a)
 UNION(d, e)
 UNION(a, e)
 UNION(f, b)
 FIND-SET(b)



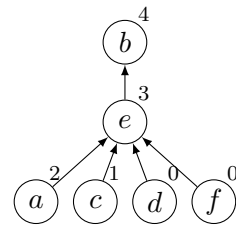
☒



☐



☐



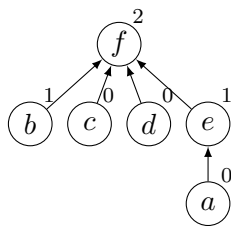
☐

Opgave/Problem 133 (4 %)

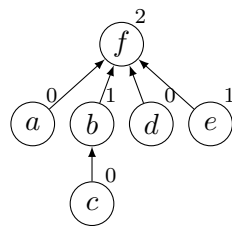
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

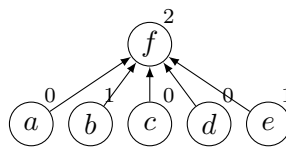
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(c, b)
 UNION(d, f)
 UNION(c, d)
 UNION(a, e)
 UNION(c, e)
 FIND-SET(a)



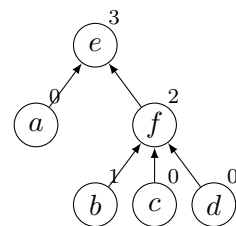
☐ A



☐ B



☒ C



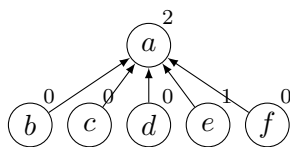
☐ D

Opgave/Problem 134 (4 %)

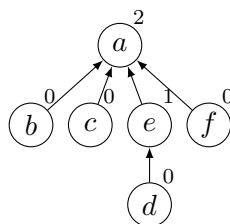
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

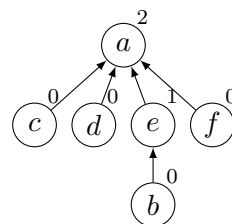
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(d, e)
 UNION(b, e)
 UNION(c, a)
 UNION(e, a)
 UNION(d, f)
 FIND-SET(b)



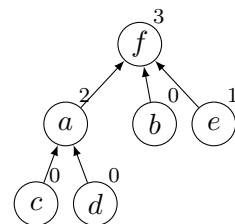
☒



☐



☐



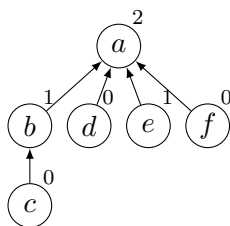
☐

Opgave/Problem 135 (4 %)

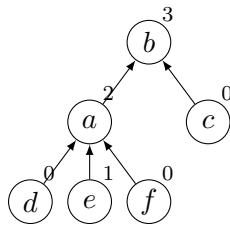
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

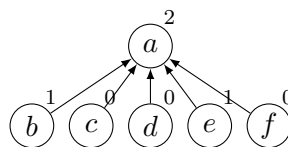
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(f, e)
 UNION(d, a)
 UNION(f, d)
 UNION(c, b)
 UNION(f, c)
 FIND-SET(b)



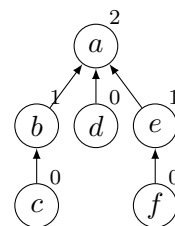
☒



☐ B



☐ C



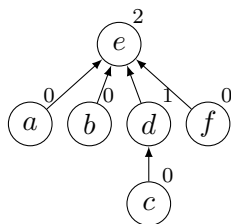
☐ D

Opgave/Problem 136 (4 %)

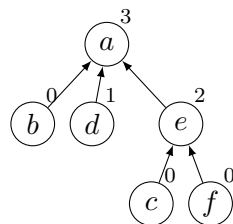
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

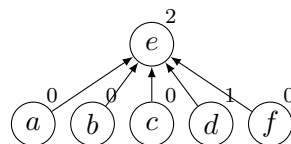
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(b , d)
 UNION(c , b)
 UNION(f , e)
 UNION(c , f)
 UNION(c , a)
 FIND-SET(b)



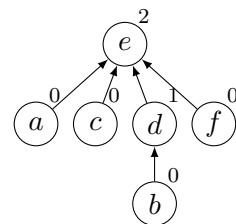
☐ A



☐ B



☒ C



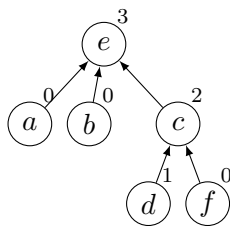
☐ D

Opgave/Problem 137 (4 %)

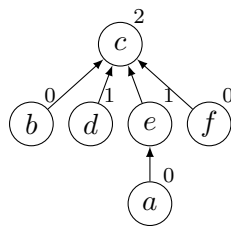
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

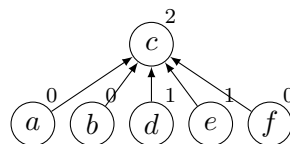
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(f , d)
 UNION(b , c)
 UNION(f , c)
 UNION(a , e)
 UNION(f , a)
 FIND-SET(b)



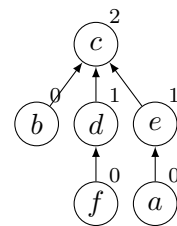
☐ A



☒ B



☐ C



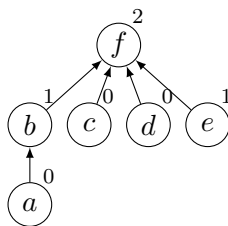
☐ D

Opgave/Problem 138 (4 %)

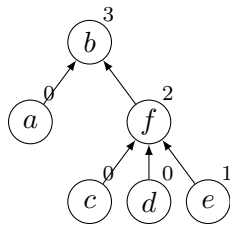
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

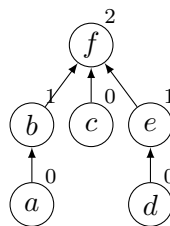
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(d, e)
 UNION(c, f)
 UNION(e, f)
 UNION(a, b)
 UNION(d, a)
 FIND-SET(b)



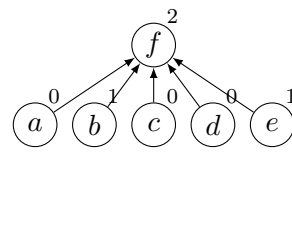
☒



☐ B



☐ C



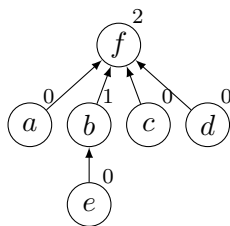
☐ D

Opgave/Problem 139 (4 %)

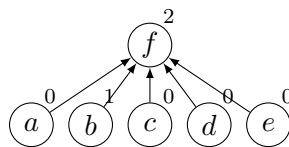
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

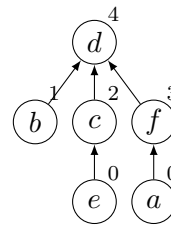
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(e, b)
 UNION(b, c)
 UNION(a, f)
 UNION(e, a)
 UNION(c, d)
 FIND-SET(b)



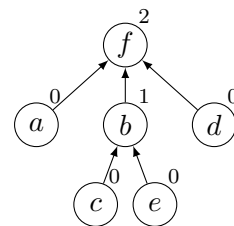
☒



☐



☐



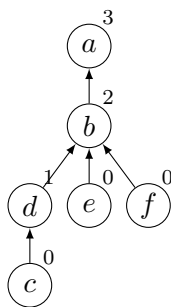
☐

Opgave/Problem 140 (4 %)

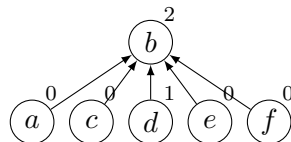
Angiv den resulterende union-find struktur efter nedenstående sekvens af operationer, når der anvendes union-by-rank og stikomprimering.

Indicate the resulting union-find structure after the following sequence of operations, when using union-by-rank and path compression.

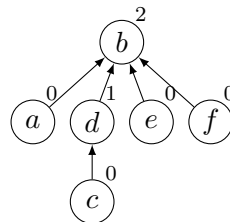
MAKESET(a)
 MAKESET(b)
 MAKESET(c)
 MAKESET(d)
 MAKESET(e)
 MAKESET(f)
 UNION(c, d)
 UNION(e, d)
 UNION(f, b)
 UNION(c, b)
 UNION(e, a)
 FIND-SET(b)



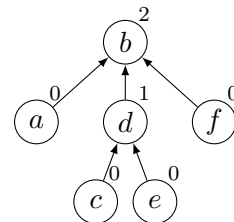
☐ A



☐ B



☒ C



☐ D

Huffman koder / *Huffman encoding*

Opgave / *Problem* 141 (4 %)

Bogstav / <i>Letter</i>	a	b	c	d	e	f	g
Frekvens / <i>Frequency</i>	20	80	10	40	70	40	30

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **a**, **d** og **f**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **a**, **d** and **f**?*

	1	2	3	4	5	6	
a	☐ A	☐ B	☐ C	☒	☐ E	☐ F	141.1
d	☐ A	☐ B	☒	☐ D	☐ E	☐ F	141.2
f	☐ A	☐ B	☒	☐ D	☐ E	☐ F	141.3

Filen indeholder $20 + 80 + 10 + 40 + 70 + 40 + 30 = 290$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $20 + 80 + 10 + 40 + 70 + 40 + 30 = 290$ letters. How many bits would a Huffman encoding of the string use?

750	790	800	810	830	840	
☒	☐ B	☐ C	☐ D	☐ E	☐ F	141.4

Opgave/Problem 142 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	60	30	50	80	80	40	30

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **b**, **d** og **g**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **b**, **d** and **g**?*

	1	2	3	4	5	6	
b	☐ A	☐ B	☐ C	☒	☐ E	☐ F	142.1
d	☐ A	☒	☐ C	☐ D	☐ E	☐ F	142.2
g	☐ A	☐ B	☐ C	☒	☐ E	☐ F	142.3

Filen indeholder $60 + 30 + 50 + 80 + 80 + 40 + 30 = 370$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $60 + 30 + 50 + 80 + 80 + 40 + 30 = 370$ letters. How many bits would a Huffman encoding of the string use?

1010 1020 1030 1040 1050 1060

☒	☐ B	☐ C	☐ D	☐ E	☐ F	142.4
-------------------------------------	----------------------------	----------------------------	----------------------------	----------------------------	----------------------------	-------

Opgave/Problem 143 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	40	10	30	20	40	70	80

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **c**, **d** og **e**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **c**, **d** and **e**?*

	1	2	3	4	5	6	
c	☐ A	☐ B	☒	☐ D	☐ E	☐ F	143.1
d	☐ A	☐ B	☐ C	☒	☐ E	☐ F	143.2
e	☐ A	☐ B	☒	☐ D	☐ E	☐ F	143.3

Filen indeholder $40 + 10 + 30 + 20 + 40 + 70 + 80 = 290$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $40 + 10 + 30 + 20 + 40 + 70 + 80 = 290$ letters. How many bits would a Huffman encoding of the string use?

730 740 750 780 790 800

☐ A	☐ B	☒	☐ D	☐ E	☐ F	143.4
----------------------------	----------------------------	-------------------------------------	----------------------------	----------------------------	----------------------------	-------

Opgave/Problem 144 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	40	20	70	60	80	50	10

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **d**, **e** og **g**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **d**, **e** and **g**?*

	1	2	3	4	5	6	
d	☐ A	☐ B	☒	☐ D	☐ E	☐ F	144.1
e	☐ A	☒	☐ C	☐ D	☐ E	☐ F	144.2
g	☐ A	☐ B	☐ C	☒	☐ E	☐ F	144.3

Filen indeholder $40 + 20 + 70 + 60 + 80 + 50 + 10 = 330$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $40 + 20 + 70 + 60 + 80 + 50 + 10 = 330$ letters. How many bits would a Huffman encoding of the string use?

	830	850	860	870	900	910	
	☐ A	☐ B	☐ C	☒	☐ E	☐ F	144.4

Opgave/Problem 145 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	70	10	50	50	10	30	20

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **a**, **c** og **f**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **a**, **c** and **f**?*

	1	2	3	4	5	6	
a	☐ A	☒	☐ C	☐ D	☐ E	☐ F	145.1
c	☐ A	☒	☐ C	☐ D	☐ E	☐ F	145.2
f	☐ A	☐ B	☒	☐ D	☐ E	☐ F	145.3

Filen indeholder $70 + 10 + 50 + 50 + 10 + 30 + 20 = 240$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $70 + 10 + 50 + 50 + 10 + 30 + 20 = 240$ letters. How many bits would a Huffman encoding of the string use?

	590	600	610	620	630	640	
	☐ A	☐ B	☒	☐ D	☐ E	☐ F	145.4

Opgave/Problem 146 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	40	80	60	20	10	40	30

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **c**, **e** og **g**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **c**, **e** and **g**?*

	1	2	3	4	5	6	
c	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	146.1
e	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	146.2
g	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	146.3

Filen indeholder $40 + 80 + 60 + 20 + 10 + 40 + 30 = 280$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $40 + 80 + 60 + 20 + 10 + 40 + 30 = 280$ letters. How many bits would a Huffman encoding of the string use?

	720	730	760	770	780	790	
	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	146.4

Opgave/Problem 147 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	70	40	50	30	20	20	30

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **c**, **d** og **f**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **c**, **d** and **f**?*

	1	2	3	4	5	6	
c	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	147.1
d	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	147.2
f	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	147.3

Filen indeholder $70 + 40 + 50 + 30 + 20 + 20 + 30 = 260$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $70 + 40 + 50 + 30 + 20 + 20 + 30 = 260$ letters. How many bits would a Huffman encoding of the string use?

	660	670	680	700	710	720	
	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	147.4

Opgave/Problem 148 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	80	50	50	60	10	40	20

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **b**, **c** og **g**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **b**, **c** and **g**?*

	1	2	3	4	5	6	
b	☐ A	☐ B	☒	☐ D	☐ E	☐ F	148.1
c	☐ A	☐ B	☒	☐ D	☐ E	☐ F	148.2
g	☐ A	☐ B	☐ C	☒	☐ E	☐ F	148.3

Filen indeholder $80 + 50 + 50 + 60 + 10 + 40 + 20 = 310$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $80 + 50 + 50 + 60 + 10 + 40 + 20 = 310$ letters. How many bits would a Huffman encoding of the string use?

	780	800	810	820	830	850	
	☐ A	☐ B	☐ C	☒	☐ E	☐ F	148.4

Opgave/Problem 149 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	50	20	70	30	30	40	20

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **b**, **d** og **e**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **b**, **d** and **e**?*

	1	2	3	4	5	6	
b	☐ A	☐ B	☐ C	☒	☐ E	☐ F	149.1
d	☐ A	☐ B	☒	☐ D	☐ E	☐ F	149.2
e	☐ A	☐ B	☒	☐ D	☐ E	☐ F	149.3

Filen indeholder $50 + 20 + 70 + 30 + 30 + 40 + 20 = 260$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $50 + 20 + 70 + 30 + 30 + 40 + 20 = 260$ letters. How many bits would a Huffman encoding of the string use?

	700	710	720	730	740	750	
	☒	☐ B	☐ C	☐ D	☐ E	☐ F	149.4

Opgave/Problem 150 (4 %)

Bogstav/Letter	a	b	c	d	e	f	g
Frekvens/Frequency	40	60	60	10	50	50	20

Antag vi har en fil med ovenstående bogstaver og frekvenser. Hvis man konstruerer et Huffman træ for disse frekvenser, hvor lange er koderne så for **b**, **e** og **f**?

*Assume we have a file with the above letters and frequencies. If one constructs a Huffman tree for these frequencies, what are the lengths of the codes for **b**, **e** and **f**?*

	1	2	3	4	5	6	
b	☐ A	☒	☐ C	☐ D	☐ E	☐ F	150.1
e	☐ A	☐ B	☒	☐ D	☐ E	☐ F	150.2
f	☐ A	☐ B	☒	☐ D	☐ E	☐ F	150.3

Filen indeholder $40 + 60 + 60 + 10 + 50 + 50 + 20 = 290$ bogstaver. Hvor mange bits vil en Huffman kodning af strengen bruge?

The file contains $40 + 60 + 60 + 10 + 50 + 50 + 20 = 290$ letters. How many bits would a Huffman encoding of the string use?

	720	730	740	760	770	780	
	☐ A	☐ B	☐ C	☐ D	☐ E	☒	150.4

Rekursionsligninger/Recurrence relations

Opgave/Problem 151 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = T(n-1) + \log n$	☐ A	☐ B	☐ C	☒	☐ E	☐ F	☐ G	151.1
$T(n) = 4 \cdot T(n/5) + n^3$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒	151.2
$T(n) = 4 \cdot T(n/2) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☒	☐ G	151.3
$T(n) = 3 \cdot T(n/5) + n$	☐ A	☐ B	☒	☐ D	☐ E	☐ F	☐ G	151.4
$T(n) = T(n-1) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒	151.5

Opgave/Problem 152 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = T(n-1) + \log n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	152.1
$T(n) = 4 \cdot T(n/5) + n$	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	152.2
$T(n) = T(n/3) + 3$	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	152.3
$T(n) = 2 \cdot T(n/2) + n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	152.4
$T(n) = 3 \cdot T(n/9) + 3$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	152.5

Opgave/Problem 153 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 2 \cdot T(n/4) + 2$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	153.1
$T(n) = 5 \cdot T(n/5) + n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	153.2
$T(n) = 3 \cdot T(n/9) + 1$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	153.3
$T(n) = 8 \cdot T(n/2) + 2$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	153.4
$T(n) = 4 \cdot T(n/2) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	153.5

Opgave/Problem 154 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ 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	☒ C	☐ D	☐ E	☐ F	☐ G	154.1
$T(n) = 4 \cdot T(n/2) + 1$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	154.2
$T(n) = 4 \cdot T(n/4) + n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	154.3
$T(n) = T(n/4) + 3$	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	154.4
$T(n) = T(n-1) + n$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	154.5

Opgave/Problem 155 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 2 \cdot T(n/4) + 1$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	155.1
$T(n) = 5 \cdot T(n/5) + n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	155.2
$T(n) = 8 \cdot T(n/2) + 1$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	155.3
$T(n) = T(n-1) + 2$	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	155.4
$T(n) = T(n-1) + n$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	155.5

Opgave/Problem 156 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 4 \cdot T(n/2) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	156.1
$T(n) = 9 \cdot T(n/3) + 3$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	156.2
$T(n) = 4 \cdot T(n/2) + 2$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	156.3
$T(n) = 2 \cdot T(n/4) + 2$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	156.4
$T(n) = 8 \cdot T(n/2) + 1$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	156.5

Opgave/Problem 157 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 8 \cdot T(n/2) + 2$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	157.1
$T(n) = T(n/3) + 5$	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	157.2
$T(n) = T(n-1) + n$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	157.3
$T(n) = 2 \cdot T(n/4) + n$	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	157.4
$T(n) = 3 \cdot T(n/4) + n^3$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	157.5

Opgave/Problem 158 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 9 \cdot T(n/3) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	158.1
$T(n) = 4 \cdot T(n/2) + 2$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	158.2
$T(n) = T(n/4) + 3$	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	158.3
$T(n) = 3 \cdot T(n/9) + 1$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	158.4
$T(n) = 3 \cdot T(n/5) + n$	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	158.5

Opgave/Problem 159 (4 %)

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

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 3 \cdot T(n/9) + 3$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	159.1
$T(n) = T(n-1) + \log n$	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	159.2
$T(n) = 2 \cdot T(n/4) + 2$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	159.3
$T(n) = T(n-1) + 3$	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	159.4
$T(n) = T(n/3) + 5$	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	159.5

Opgave/Problem 160 (4 %)

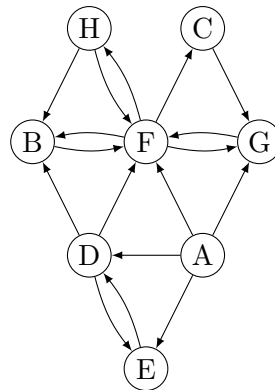
Angiv løsningen for hver af nedenstående rekursionsligninger, hvor $T(n) = 1$ for $n \leq 1$.

State the solution for each of the below recurrence relations, where $T(n) = 1$ for $n \leq 1$.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	$\Theta(n^3)$	
$T(n) = 9 \cdot T(n/3) + 3$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	160.1
$T(n) = 4 \cdot T(n/5) + n^2$	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	160.2
$T(n) = 4 \cdot T(n/2) + n^2$	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	160.3
$T(n) = 3 \cdot T(n/9) + 2$	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	160.4
$T(n) = 2 \cdot T(n/4) + n^3$	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	160.5

BFS

Opgave/Problem 161 (4 %)



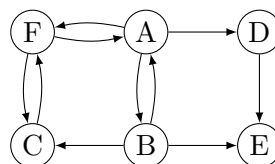
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver udtaget af køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ADEFG BCH ADEFG BHC AGDEFBCH ADBFCGHE



Opgave/Problem 162 (4 %)



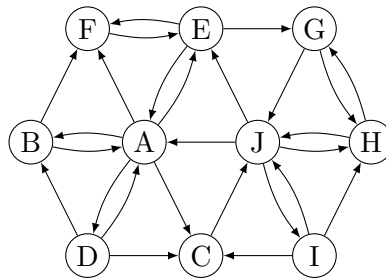
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver indsat i køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ABCFED ABDFCE ABDFEC AFDBCE



Opgave/Problem 163 (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 alfabetisk sorterede incidenslister.

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

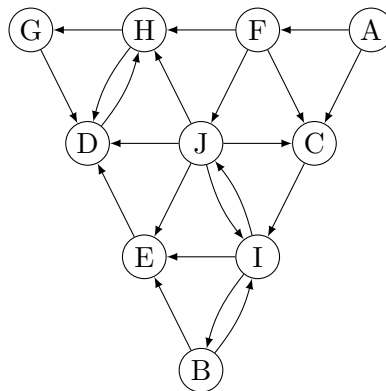
ABFEGHJICD ABCDEFJGHIH AEBFDCGJHI ABCDEFJGHI

A

B

C

Opgave/*Problem* 164 (4 %)



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

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ACIBEDHGJF ACFIHJBEGD ACFIHJEBGD ACFIHJBEDG

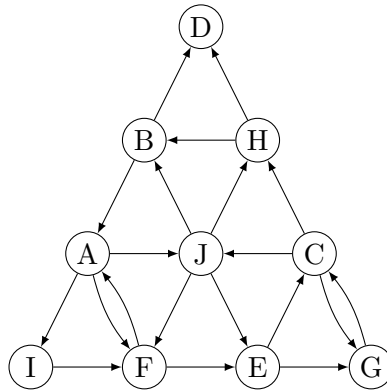
A

B

C



Opgave/Problem 165 (4 %)



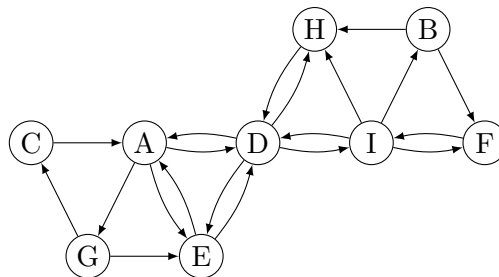
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver indsat i køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

*For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.*

AFIJEHBHCGD AJFIHBEDCG AFIJEBHCGD AFECGHBDJI



Opgave/Problem 166 (4 %)



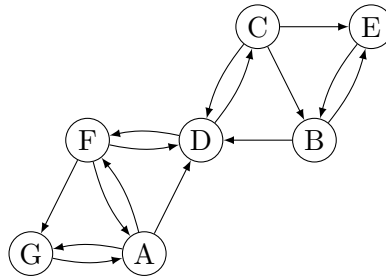
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver udtaget af køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

*For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.*

ADEGHICBF ADEHIBFGC ADEGHICFB ADEGIHCFB



Opgave/Problem 167 (4 %)



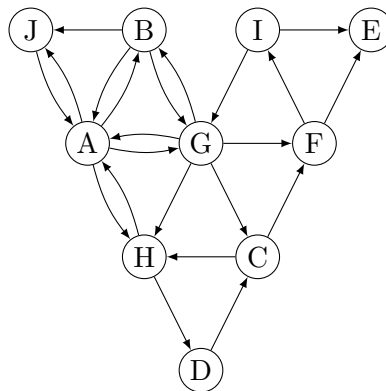
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver udtaget af køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are removed from the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ADFGCBE ADGFCEB ADFGCEB ADCBEFG



Opgave/Problem 168 (4 %)



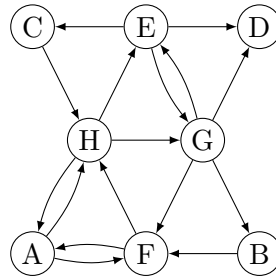
For et bredde først gennemløb (BFS) af ovenstående graf **startende i knuden A**, angiv rækkefølgen knuderne bliver udtaget af køen Q i BFS-algoritmen. Det antages, at grafen er givet ved alfabetisk sorterede incidenslister.

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are removed from the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ABGCFEIHJ ABGHJCFDIE ABGHJCFDEI ABHJGDCFEI



Opgave/Problem 169 (4 %)



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

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

AFHECDGB AHFGEBDC AFHEGDCB AFHEGDCB

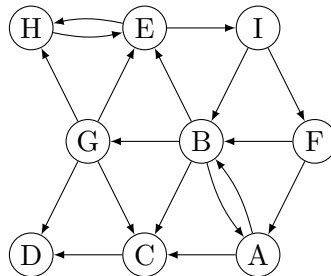
☐ A

☐ B

☐ C

☒

Opgave/Problem 170 (4 %)



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

For a breadth first traversal (BFS) of the graph above **starting in node A**, indicate the order in which the nodes are inserted into the queue Q in the BFS algorithm. It is assumed that the graph is given by alphabetically sorted adjacency lists.

ABCDEHIFG ACBDGEHIF ABCEGDHIF ABCEGDIHF

☐ A

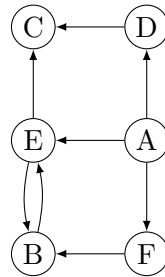
☐ B

☒

☐ D

Lovlige BFS træer/ *Valid BFS trees*

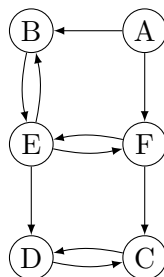
Opgave/Problem 171 (4 %)



Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

*Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.*

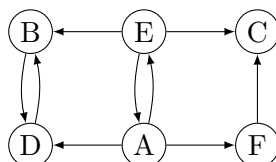
	Ja/Yes	Nej/No	
(A,D) (A,F) (B,E) (E,C) (F,B)	☐	☒	171.1
(A,D) (A,E) (A,F) (E,B) (E,C)	☒	☐	171.2
(A,D) (A,E) (A,F) (D,C) (E,B)	☒	☐	171.3
(A,D) (A,F) (B,E) (D,C) (F,B)	☐	☒	171.4
(A,D) (A,E) (A,F) (E,C) (F,B)	☒	☐	171.5

Opgave/Problem 172 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

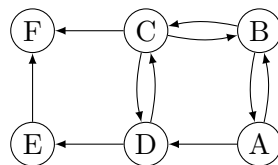
	Ja/Yes	Nej/No	
(A,B) (A,F) (E,D) (F,C) (F,E)	☒	☐	172.1
(A,B) (B,E) (D,C) (E,D) (E,F)	☐	☒	172.2
(A,B) (A,F) (D,C) (E,D) (F,E)	☐	☒	172.3
(A,F) (E,B) (E,D) (F,C) (F,E)	☐	☒	172.4
(A,B) (A,F) (C,D) (F,C) (F,E)	☒	☐	172.5

Opgave/Problem 173 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

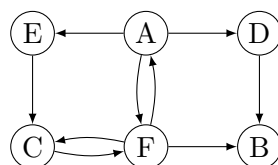
	Ja/Yes	Nej/No	
(A,D) (A,E) (A,F) (D,B) (F,C)	☒	☐	173.1
(A,E) (A,F) (B,D) (E,B) (E,C)	☐	☒	173.2
(A,D) (A,E) (A,F) (E,B) (F,C)	☒	☐	173.3
(A,D) (A,E) (A,F) (E,B) (E,C)	☒	☐	173.4
(A,D) (A,E) (A,F) (D,B) (E,C)	☒	☐	173.5

Opgave/Problem 174 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

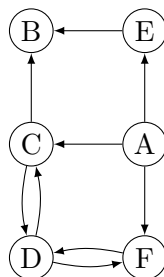
	Ja/Yes	Nej/No	
(A,B) (A,D) (C,F) (D,C) (D,E)	☒	☐	174.1
(A,D) (C,B) (D,C) (D,E) (E,F)	☐	☒	174.2
(A,B) (A,D) (D,C) (D,E) (E,F)	☒	☐	174.3
(A,B) (A,D) (B,C) (C,F) (D,E)	☒	☐	174.4
(A,B) (A,D) (B,C) (D,E) (E,F)	☐	☒	174.5

Opgave/Problem 175 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

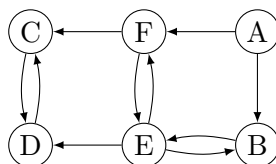
	Ja/Yes	Nej/No	
(A,D) (A,E) (A,F) (D,B) (F,C)	☒	☐	175.1
(A,D) (A,E) (A,F) (D,B) (E,C)	☒	☐	175.2
(A,D) (A,E) (A,F) (E,C) (F,B)	☒	☐	175.3
(A,D) (A,E) (C,F) (E,C) (F,B)	☐	☒	175.4
(A,D) (A,E) (A,F) (F,B) (F,C)	☒	☐	175.5

Opgave/Problem 176 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

	Ja/Yes	Nej/No	
(A,C) (A,E) (A,F) (C,D) (E,B)	☒	☐	176.1
(A,C) (A,E) (A,F) (E,B) (F,D)	☒	☐	176.2
(A,E) (A,F) (C,B) (D,C) (F,D)	☐	☒	176.3
(A,C) (A,E) (A,F) (C,B) (C,D)	☒	☐	176.4
(A,C) (A,E) (A,F) (C,B) (F,D)	☒	☐	176.5

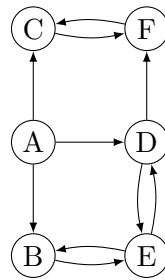
Opgave/Problem 177 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

	Ja/Yes	Nej/No	
(A,B) (A,F) (E,D) (F,C) (F,E)	☒	☐	177.1
(A,B) (B,E) (D,C) (E,D) (E,F)	☐	☒	177.2
(A,B) (A,F) (B,E) (E,D) (F,C)	☒	☐	177.3
(A,F) (C,D) (E,B) (F,C) (F,E)	☐	☒	177.4
(A,B) (A,F) (B,E) (D,C) (E,D)	☐	☒	177.5

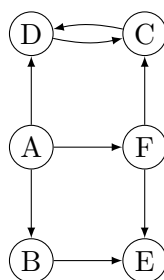
Opgave/Problem 178 (4 %)



Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

*Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.*

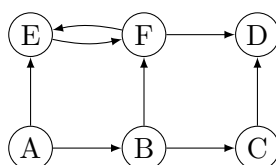
	Ja/ Yes	Nej/ No	
(A,D) (D,E) (D,F) (E,B) (F,C)	☐	☒	178.1
(A,B) (A,C) (A,D) (C,F) (D,E)	☒	☐	178.2
(A,B) (A,C) (A,D) (B,E) (C,F)	☒	☐	178.3
(A,B) (A,C) (B,E) (C,F) (E,D)	☐	☒	178.4
(A,B) (A,C) (A,D) (D,E) (D,F)	☒	☐	178.5

Opgave/Problem 179 (4 %)


Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

	Ja/Yes	Nej/No	
(A,B) (A,F) (C,D) (F,C) (F,E)	☐	☒	179.1
(A,B) (A,D) (A,F) (B,E) (F,C)	☒	☐	179.2
(A,B) (A,D) (A,F) (F,C) (F,E)	☒	☐	179.3
(A,B) (A,D) (A,F) (B,E) (D,C)	☒	☐	179.4
(A,B) (A,D) (A,F) (D,C) (F,E)	☒	☐	179.5

Opgave/Problem 180 (4 %)


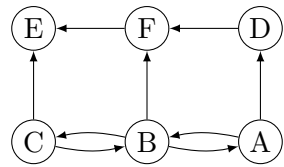
Angiv for hver af nedenstående mængder af kanter om de udgør et lovligt BFS træ for et bredde først gennemløb af ovenstående graf **startende i knuden A** og for en passende ordning af grafens incidenslister.

Indicate for each of the sets of edges below whether they form a valid BFS tree for a breadth first traversal of the graph above **starting in node A** and for a suitable ordering of the graph's adjacency lists.

	Ja/Yes	Nej/No	
(A,B) (B,C) (B,F) (F,D) (F,E)	☐	☒	180.1
(A,B) (A,E) (B,C) (C,D) (E,F)	☐	☒	180.2
(A,B) (A,E) (B,C) (B,F) (C,D)	☒	☐	180.3
(A,B) (A,E) (B,C) (E,F) (F,D)	☒	☐	180.4
(A,B) (A,E) (B,C) (B,F) (F,D)	☒	☐	180.5

DFS

Opgave/*Problem* 181 (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 **discovery time**.

*Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.*

ABCEFD ADBFCE ABDCFE ADFEBC

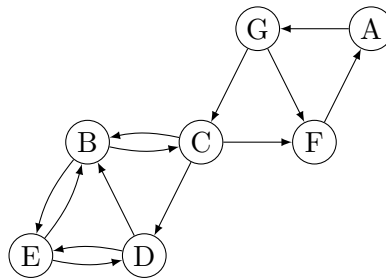
☒ ☐ ☐ ☐ 181.1

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

Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(A, D)	☒	☐	☐	☐	181.2
(C, B)	☐	☒	☐	☐	181.3
(F, E)	☐	☐	☒	☐	181.4
(B, A)	☐	☒	☐	☐	181.5

Opgave/Problem 182 (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 **discovery time**.

Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.

AGCFBED AGCBEDF AGCFBDE AGFCBED

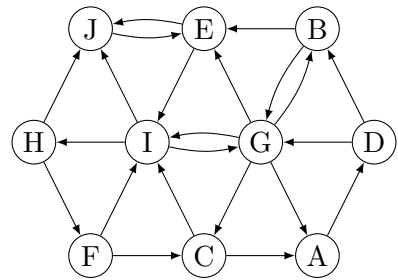
☐ A ☒ B ☐ C ☐ D 182.1

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

Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(C, D)	☐ A	☐ B	☐ C	☒ D	182.2
(D, E)	☐ A	☒ B	☐ C	☐ D	182.3
(D, B)	☐ A	☒ B	☐ C	☐ D	182.4
(C, B)	☒ A	☐ B	☐ C	☐ D	182.5

Opgave/Problem 183 (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 **discovery time**.

*Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.*

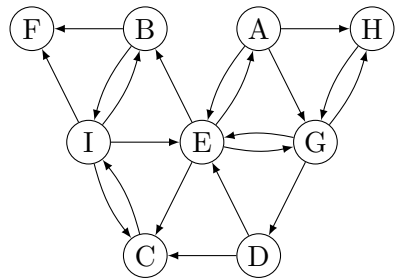
ADBGEICIJHF ADBEIHJFGC ADBEIGCHFJ ADGBEJIHFC

☐ A ☐ B ☒ C ☐ D 183.1

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet.
Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(F, C)	☐ A	☐ B	☒ C	☐ D	183.2
(E, J)	☐ A	☐ B	☐ C	☒ D	183.3
(F, I)	☐ A	☒ B	☐ C	☐ D	183.4
(B, E)	☒ A	☐ B	☐ C	☐ D	183.5

Opgave/Problem 184 (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**.

*Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **finishing time**.*

FCIBDHGEA HDGCIFBEA FBICDHGEA IFDCBHGEA



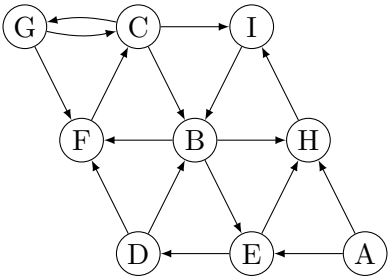
184.1

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

Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(D, E)	☐	☒	☐	☐	184.2
(A, H)	☐	☐	☐	☒	184.3
(A, E)	☒	☐	☐	☐	184.4
(I, F)	☐	☐	☒	☐	184.5

Opgave/Problem 185 (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 **discovery time**.

Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.

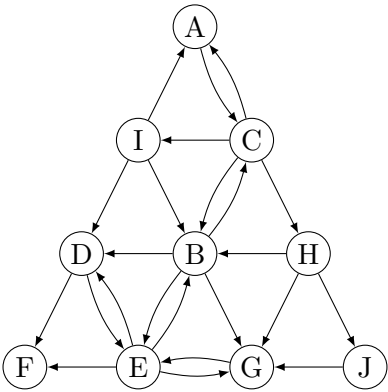
AEDBHFCIG AEDBHIFCG AEDBFCGIH AEHDIBFCG

☐ A ☐ B ☒ C ☐ D 185.1

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet. Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(D, F)	☐ A	☐ B	☐ C	☒ D	185.2
(I, B)	☐ A	☒ B	☐ C	☐ D	185.3
(B, H)	☒ A	☐ B	☐ C	☐ D	185.4
(H, I)	☐ A	☐ B	☒ C	☐ D	185.5

Opgave/Problem 186 (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**.

Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **finishing time**.

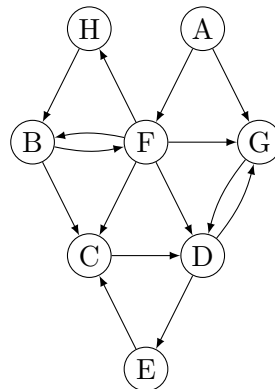
I J H G F E D B C A F G B E D I J H C A F J G E D I H B C A F G E D B J H I C A

☐ A ☐ B ☐ C ☒ 186.1

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet.
Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(D, F)	☐ A	☐ B	☐ C	☒ D	186.2
(D, E)	☒ A	☐ B	☐ C	☐ D	186.3
(I, A)	☐ A	☒ B	☐ C	☐ D	186.4
(I, D)	☐ A	☐ B	☒ C	☐ D	186.5

Opgave/Problem 187 (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**.

Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **finishing time**.

EGDCBHFA EHD CBGFA HGEDCBFA GEDCBHFA



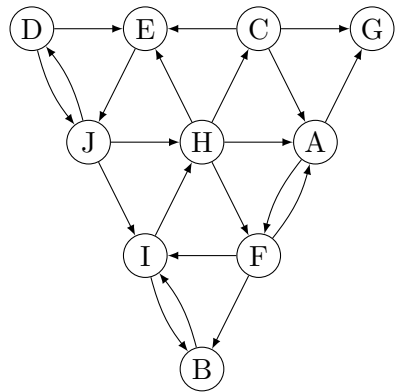
187.1

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

Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(G, D)	☐	☒	☐	☐	187.2
(H, B)	☐	☐	☒	☐	187.3
(D, G)	☒	☐	☐	☐	187.4
(A, G)	☐	☐	☐	☒	187.5

Opgave/Problem 188 (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 **discovery time**.

Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.

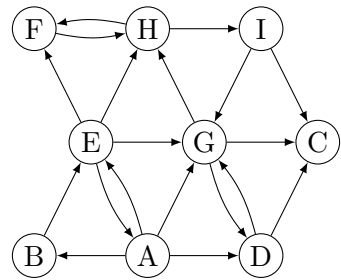
AFGBIHCEJD AFBHICGEJD AFBIHCEJDG AGFIHCEJDB

☐ A ☐ B ☒ C ☐ D 188.1

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet.
Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(H, C)	☒	☐	☐	☐	188.2
(D, J)	☐	☒	☐	☐	188.3
(F, I)	☐	☐	☐	☒	188.4
(I, B)	☐	☒	☐	☐	188.5

Opgave/Problem 189 (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 **discovery time**.

*Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **discovery time**.*

ABEFHICGD ABDEGCFHI ABEFHIGDC AEFHIGCDB



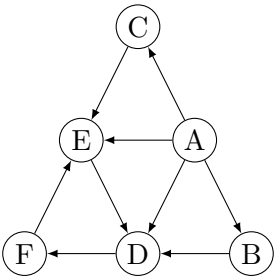
189.1

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

Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(E, H)	☐	☐	☐	☒	189.2
(G, C)	☐	☐	☒	☐	189.3
(E, F)	☒	☐	☐	☐	189.4
(H, F)	☐	☒	☐	☐	189.5

Opgave/Problem 190 (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**.

*Consider a depth first traversal (DFS) of the graph above, where the DFS traversal starts in **node A**, and where the outgoing edges of a node are visited in alphabetical order. Indicate the order in which the nodes receive their **finishing time**.*

CEFDBA EFDBCA FDEBCA FEDCBA

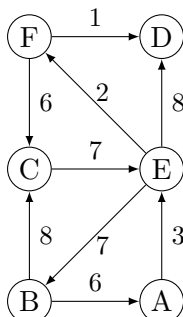
☐A ☒B ☐C ☐D 190.1

Angiv for hver af nedenstående kanter hvilken type kanten bliver i DFS gennemløbet.
Indicate for each of the edges below which type the edge is in the DFS traversal.

	Tree edge	Back edge	Cross edge	Forward edge	
(F, E)	☒	☐ B	☐ C	☐ D	190.2
(E, D)	☐ A	☒	☐ C	☐ D	190.3
(C, E)	☐ A	☐ B	☒	☐ D	190.4
(A, E)	☐ A	☐ B	☐ C	☒	190.5

Dijkstras algoritme/ *Dijkstra's algorithm*

Opgave/ *Problem 191* (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

AEBCDF AEBDFC AEFDCB AEFDBC

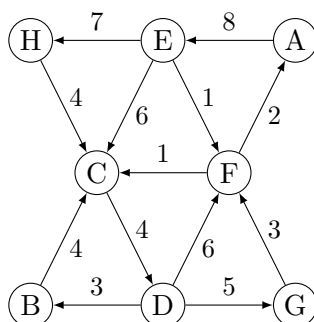
☐ A

☐ B

☐ C

☒

Opgave/ *Problem 192* (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

AECD BFGH AEFCD BFGH AECFH D B G AEFCD H B G

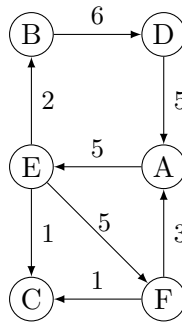
☐ A

☐ B

☐ C

☒

Opgave/Problem 193 (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 Dijkstras algoritme.

*Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.*

AEBCFD AECBFD AECBDF AEBDCF

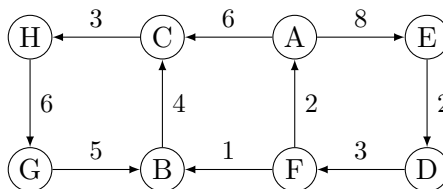
☐ A

☒ B

☐ C

☐ D

Opgave/Problem 194 (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 Dijkstras algoritme.

*Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.*

ACEHGBDF ACEHDGFB ACEHDFBG ACHGBEDF

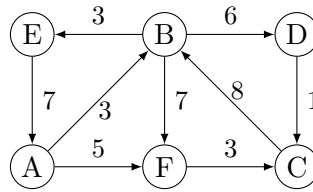
☐ A

☐ B

☒ C

☐ D

Opgave/Problem 195 (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 Dijkstras algoritme.

*Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.*

ABFCED ABDCEF ABFECD ABFDEC

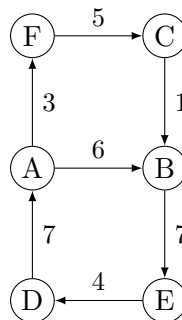
☐ A

☐ B

☒ C

☐ D

Opgave/Problem 196 (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 Dijkstras algoritme.

*Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.*

AFBCED AFBEDC ABFECD ABEDFC

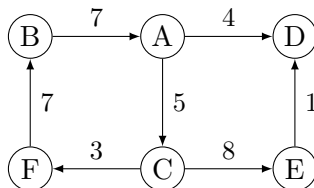
☒ A

☐ B

☐ C

☐ D

Opgave/Problem 197 (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

ADCFBE ADCFEB ACDEFB ACEDFB

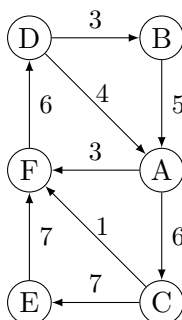
☐ A

☒ B

☐ C

☐ D

Opgave/Problem 198 (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

ACEFDB AFCEDB ACFEDB AFCDBE

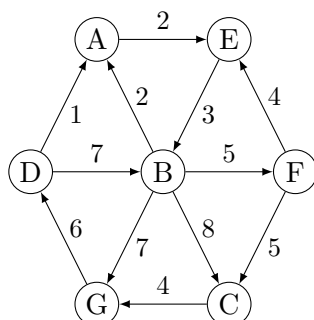
☐ A

☐ B

☐ C

☒ D

Opgave/Problem 199 (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

AEBCFGD AEBCGDF AEBFCGD AEBFGCD

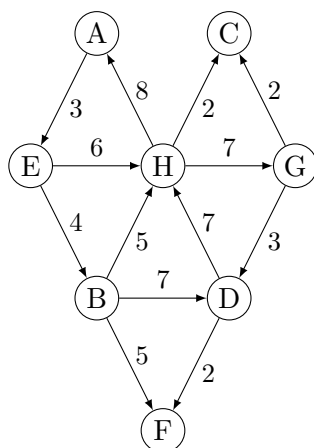
☐ A

☐ B

☐ C

☒ D

Opgave/Problem 200 (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 Dijkstras algoritme.

Assume Dijkstra's algorithm is used to find shortest distances from **node A** to all nodes in the graph above. Indicate the order in which the nodes are removed from the priority queue in Dijkstra's algorithm.

AEBHCGDF AEBHCFDG AEBDFHCG AEBHDFCG

☐ A

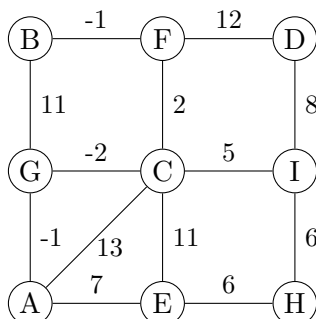
☒ B

☐ C

☐ D

Prims algoritme/*Prim's algorithm*

Opgave/*Problem* 201 (4 %)



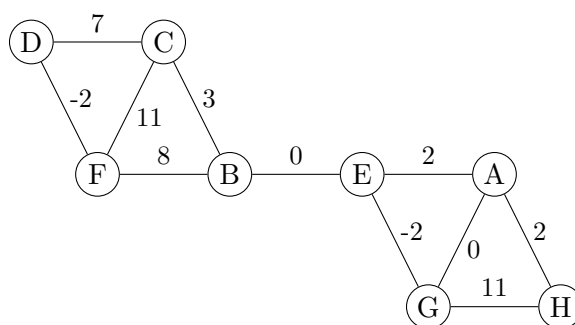
Antag Prims algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf, og algoritmen starter i **knuden A**. Angiv hvilken rækkefølge knuderne bliver inkluderet i det minimum udspændende træ (taget ud af prioritetskøen i Prims algoritme).

Assume *Prim's algorithm* is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in *Prim's algorithm*).

AGCFBIHED AGCFBDIHE AGCFBIHDE AGCFBIEHD



Opgave/*Problem* 202 (4 %)



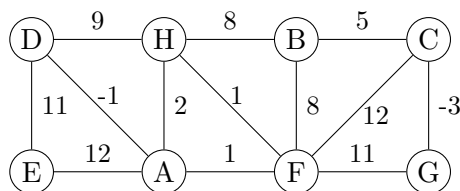
Antag Prims algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf, og algoritmen starter i **knuden A**. Angiv hvilken rækkefølge knuderne bliver inkluderet i det minimum udspændende træ (taget ud af prioritetskøen i Prims algoritme).

Assume *Prim's algorithm* is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in *Prim's algorithm*).

AGEBHCDF AGEBCHFD AGEBCDFH AGEBCHDF



Opgave/Problem 203 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

ADFBHEGC ADFHBCGE ADFHBECG ADHFBCGE

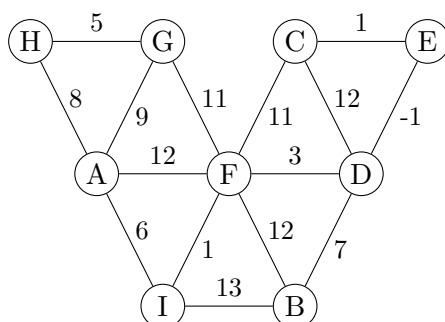
☐ A

☒ B

☐ C

☐ D

Opgave/Problem 204 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

AIFDECBHG AIFHGDECB AIFDECBGH AIFHDECGB

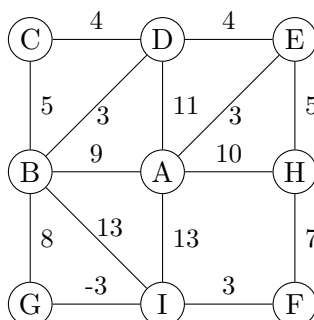
☒ A

☐ B

☐ C

☐ D

Opgave/Problem 205 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

AEDBCGIFH AEDHBCIGF AEDBCHFIG AEDHBCFIG

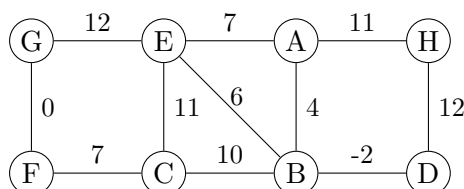
☐ A

☐ B

☒ C

☐ D

Opgave/Problem 206 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

ABDEHCGF ABDHECFG ABDEHCFG ABDECFGH

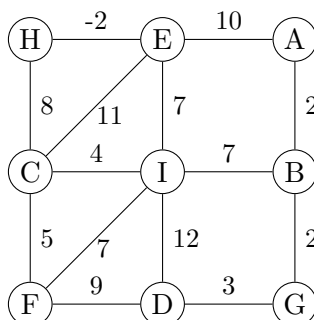
☐ A

☐ B

☐ C

☒ D

Opgave/Problem 207 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

ABGDFCIEH ABGDIEHCF ABGDICFEH ABGDICEHF

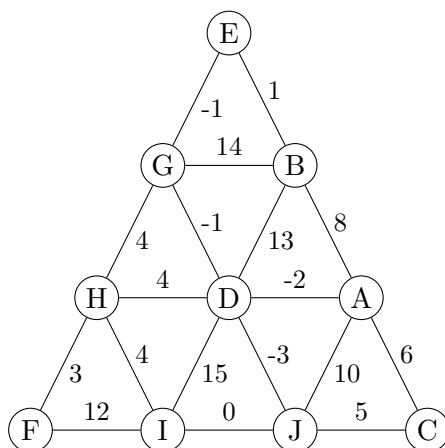
☐ A

☐ B

☒ C

☐ D

Opgave/Problem 208 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

ADJIHFGEB ADJGEIBHFC ADJIGEBCHF ADJIGEBHCF

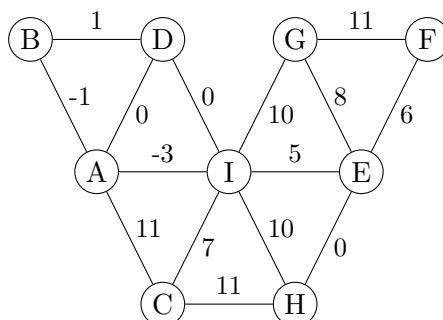
☐ A

☒ B

☐ C

☐ D

Opgave/Problem 209 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

AIBDEHCFG AIDBEHCGF AIDBEHCFG AIBDEHFCG

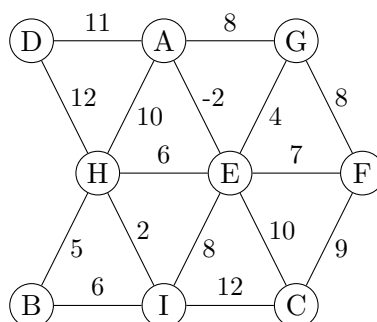
☐ A

☐ B

☐ C

☒

Opgave/Problem 210 (4 %)



Antag Prim's 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 Prim's algoritme).

Assume Prim's algorithm is used to find a minimum spanning tree for the graph above, and the algorithm starts in **node A**. Indicate the order in which the nodes are included in the minimum spanning tree (taken out of the priority queue in Prim's algorithm).

AEGHFIBDC AEGFCIHBD AEGHIBFCD AEGHFICBD

☐ A

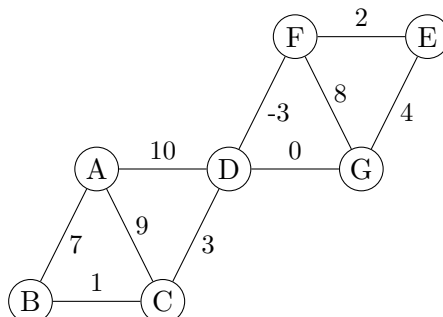
☐ B

☒

☐ D

Kruskals algoritme/ *Kruskal's algorithm*

Opgave/ *Problem 211* (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(B, C) (A, B) (E, G) (E, F) (D, G)

☐ A

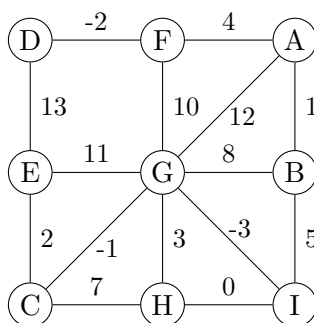
☒ B

☐ C

☐ D

☐ E

Opgave/ *Problem 212* (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(B, I) (C, H) (C, E) (H, I) (G, I)

☒ A

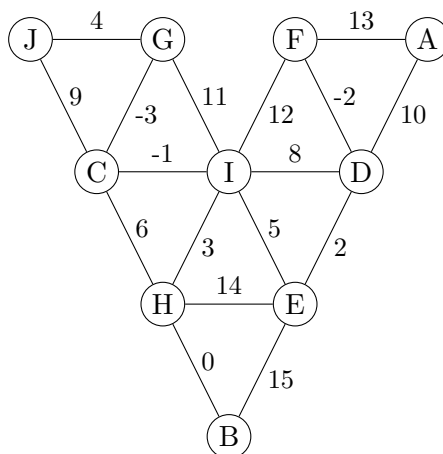
☐ B

☐ C

☐ D

☐ E

Opgave/Problem 213 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(B, H) (B, E) (A, D) (C, I) (G, J)

☐ A

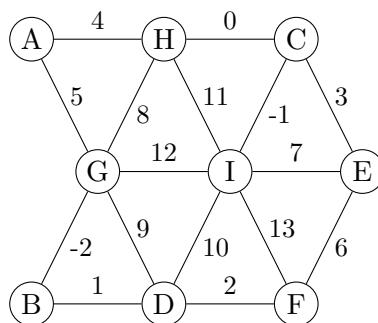
☐ B

☒ C

☐ D

☐ E

Opgave/Problem 214 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(B, G) (A, G) (C, E) (A, H) (D, F)

☐ A

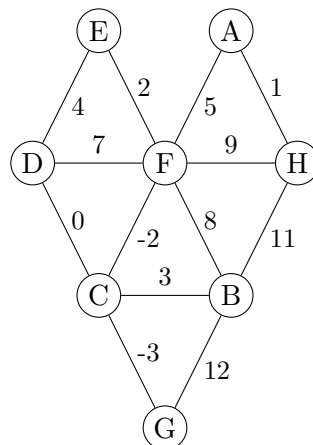
☒ B

☐ C

☐ D

☐ E

Opgave/Problem 215 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(D, E) (A, F) (E, F) (C, F) (B, C)

☐ A

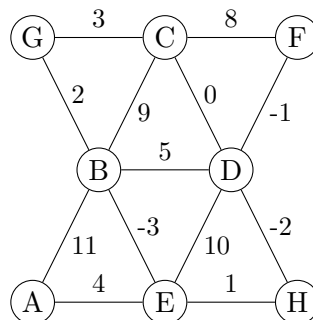
☒ B

☐ C

☐ D

☐ E

Opgave/Problem 216 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(D, F) (C, D) (A, E) (B, G) (B, E)

☐ A

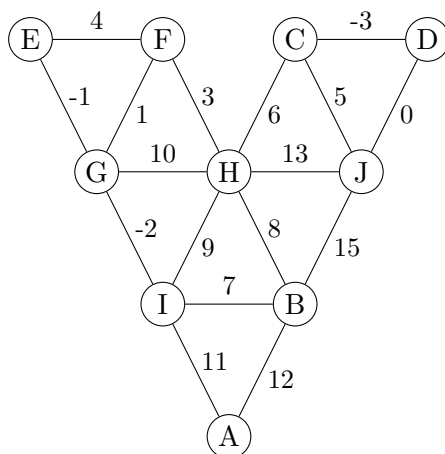
☐ B

☒ C

☐ D

☐ E

Opgave/Problem 217 (4 %)



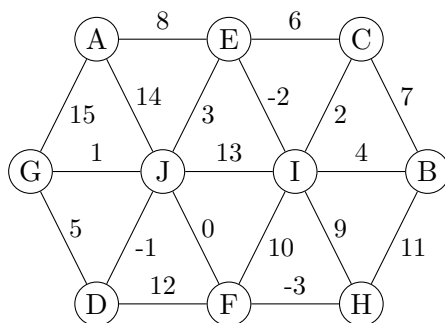
Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(B,I) (C,H) (D,J) (B,J) (A,I)

☐ A ☐ B ☐ C ☐ D ☒

Opgave/Problem 218 (4 %)



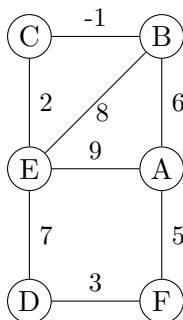
Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(D,G) (B,I) (C,I) (A,E) (G,J)

☐ A ☐ B ☐ C ☒ D ☐ E

Opgave/Problem 219 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(A, F) (B, C) (D, F) (C, E) (A, B)

☐ A

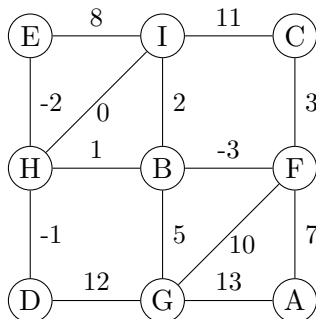
☐ B

☐ C

☐ D

☒

Opgave/Problem 220 (4 %)



Antag Kruskals algoritme anvendes til at finde et minimum udspændende træ for ovenstående graf. Angiv hvilken kant **sidst** inkluderes i det minimum udspændende træ.

Assume Kruskal's algorithm is used to find a minimum spanning tree for the graph above. Indicate which edge is **last** included in the minimum spanning tree.

(C, F) (B, G) (B, H) (D, G) (A, F)

☐ A

☐ B

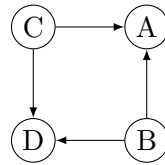
☐ C

☐ D

☒

Topologisk sortering/ *Topological sorting*

Opgave/ *Problem 221* (4 %)

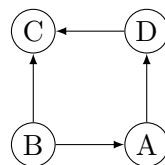


Angiv for hver af nedenstående ordninger af knuderne i ovenstående graf om det er en lovlig topologisk sortering.

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

	Ja/ <i>Yes</i>	Nej/ <i>No</i>	
B C D A	☒	☐	221.1
C B A D	☒	☐	221.2
C D A B	☐	☒	221.3
B C A D	☒	☐	221.4
D B A C	☐	☒	221.5

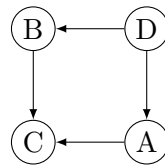
Opgave/ *Problem 222* (4 %)



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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

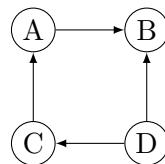
	Ja/ <i>Yes</i>	Nej/ <i>No</i>	
A B D C	☐	☒	222.1
C A D B	☐	☒	222.2
B D A C	☐	☒	222.3
B A D C	☒	☐	222.4
B A C D	☐	☒	222.5

Opgave/Problem 223 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

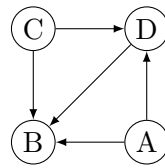
	Ja/Yes	Nej/No	
D C A B	☐	☒	223.1
D A B C	☒	☐	223.2
D B A C	☒	☐	223.3
C A B D	☐	☒	223.4
A D B C	☐	☒	223.5

Opgave/Problem 224 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

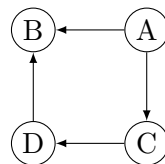
	Ja/Yes	Nej/No	
D C B A	☐	☒	224.1
D C A B	☒	☐	224.2
D A C B	☐	☒	224.3
B C A D	☐	☒	224.4
C D A B	☐	☒	224.5

Opgave/Problem 225 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

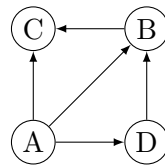
	Ja/Yes	Nej/No	
A C D B	☒	☐ B	225.1
B C D A	☐ A	☒	225.2
C A B D	☐ A	☒	225.3
C A D B	☒	☐ B	225.4
C D A B	☐ A	☒	225.5

Opgave/Problem 226 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

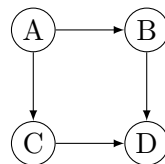
	Ja/Yes	Nej/No	
A C D B	☒	☐ B	226.1
A D C B	☐ A	☒	226.2
A B D C	☐ A	☒	226.3
D C A B	☐ A	☒	226.4
C A D B	☐ A	☒	226.5

Opgave/Problem 227 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

	Ja/Yes	Nej/No	
A D C B	☐	☒	227.1
A D B C	☒	☐	227.2
C D B A	☐	☒	227.3
A C B D	☐	☒	227.4
D A B C	☐	☒	227.5

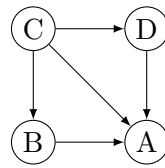
Opgave/Problem 228 (4 %)


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

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

	Ja/Yes	Nej/No	
B A C D	☐	☒	228.1
C B A D	☐	☒	228.2
A C B D	☒	☐	228.3
A D B C	☐	☒	228.4
A B C D	☒	☐	228.5

Opgave/Problem 229 (4 %)

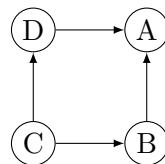


Angiv for hver af nedenstående ordninger af knuderne i ovenstående graf om det er en lovlig topologisk sortering.

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

	Ja/Yes	Nej/No	
D B C A	☐	☒	229.1
C B A D	☐	☒	229.2
A B D C	☐	☒	229.3
C B D A	☒	☐	229.4
C D B A	☒	☐	229.5

Opgave/Problem 230 (4 %)



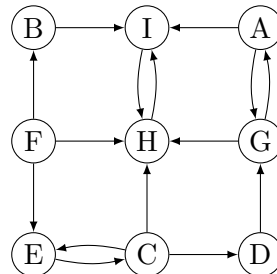
Angiv for hver af nedenstående ordninger af knuderne i ovenstående graf om det er en lovlig topologisk sortering.

For each of the following orderings of the nodes in the graph above, indicate whether it is a valid topological sort.

	Ja/Yes	Nej/No	
C D B A	☒	☐	230.1
C B D A	☒	☐	230.2
C D A B	☐	☒	230.3
C B A D	☐	☒	230.4
D B C A	☐	☒	230.5

Stærke sammenhængskomponenter / *Strongly connected components*

Opgave/Problem 231 (4 %)



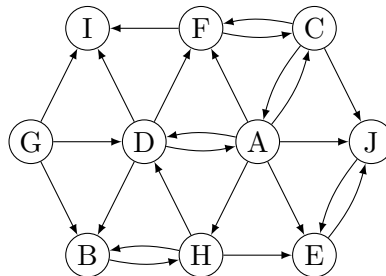
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☐ B ☐ C ☐ D ☐ E ☒ F ☐ G ☐ H ☐ I

Opgave/Problem 232 (4 %)



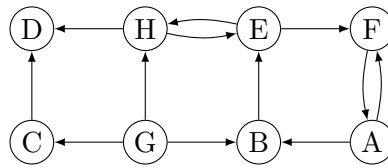
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9 10

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

Opgave/Problem 233 (4 %)



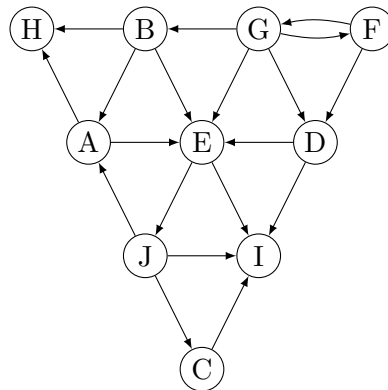
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8

☐ A ☐ B ☐ C ☒ D ☐ E ☐ F ☐ G ☐ H

Opgave/Problem 234 (4 %)



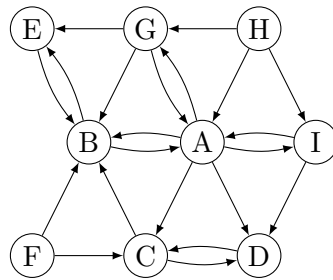
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9 10

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

Opgave/Problem 235 (4 %)



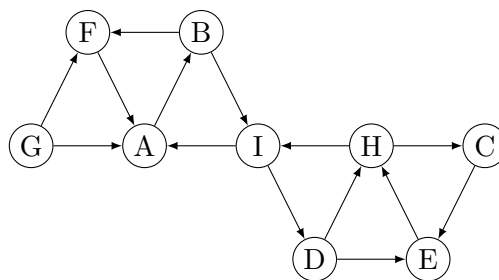
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☐ B ☒ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I

Opgave/Problem 236 (4 %)



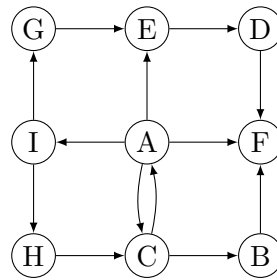
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☒ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I

Opgave/Problem 237 (4 %)



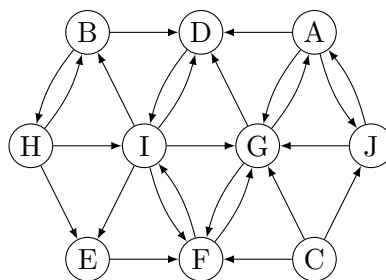
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☐ B ☐ C ☐ D ☐ E ☒ F ☐ G ☐ H ☐ I

Opgave/Problem 238 (4 %)



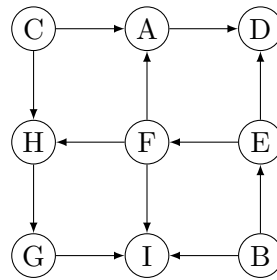
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9 10

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

Opgave/Problem 239 (4 %)



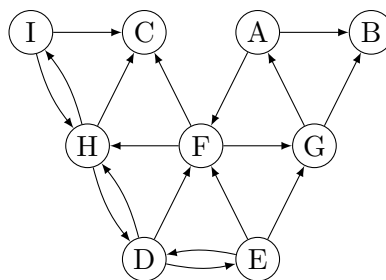
Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☒

Opgave/Problem 240 (4 %)



Hvad er antallet af stærke sammenhængskomponenter i ovenstående graf?

What is the number of strongly connected components in the graph above?

1 2 3 4 5 6 7 8 9

☐ A ☐ B ☒ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I

Løkke opgaver/ *Loop problems*

Opgave/ *Problem 241*

Algorithm loop1(n) $i = 1$ while $i \leq n$ $j = 1$ while $j \leq i$ $j = 2 * j$ $i = 2 * i$	Algorithm loop2(n) $i = 1$ while $i \leq n$ $j = 1$ while $j \leq n$ $j = 2 * j$ $i = 2 * i$	Algorithm loop3(n) $i = 1$ while $i \leq n$ $j = i$ while $j \leq n$ $j = 2 * j$ $i = 2 * i$
Algorithm loop4(n) $i = 1$ while $i \leq n$ $j = i$ while $j > 0$ $j = \lfloor j/2 \rfloor$ $i = i + i$	Algorithm loop5(n) $i = n$ while $i > 0$ $j = i$ while $j > 0$ $j = \lfloor j/2 \rfloor$ $i = \lfloor i/2 \rfloor$	Algorithm loop6(n) $s = 1$ for $i = 1$ to n $j = 1$ while $j \leq s$ $j = j + 1$ $s = 2 * s$
Algorithm loop7(n) $s = 1$ for $i = 1$ to n $j = s$ while $j > 0$ $s = s + 1$ $j = j - 1$	Algorithm loop8(n) $i = 1$ $p = 1$ while $p \leq n$ $i = i + 1$ $p = p * i$	Algorithm loop9(n) $i = 1$ $j = n$ while $i < j$ $i = 2 * i$ $j = j + n$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(2^n)$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(\log n)$	$\Theta(\frac{\log n}{\log \log n})$	$\Theta(n \log n)$	$\Theta(n)$	$\Theta((\log n)^2)$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	241.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	241.2
loop3	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	241.3
loop4	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	241.4
loop5	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒ H	241.5
loop6	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	241.6
loop7	☒ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	241.7
loop8	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	241.8
loop9	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	241.9

Opgave/Problem 242

Algorithm loop1(n) $i = 1$ $j = n$ while $i \leq j$ $\quad i = 4 * i$ $\quad j = 2 * j$	Algorithm loop2(n) $i = 1$ $j = n$ while $i \leq j$ $\quad i = i * 2$ $\quad j = \lfloor j/2 \rfloor$	Algorithm loop3(n) $i = 1$ while $i * i \leq n$ $\quad i = i + i$
Algorithm loop4(n) $i = 1$ while $i \leq n$ $\quad i = 2 * i$	Algorithm loop5(n) $i = 1$ while $i \leq n$ $\quad i = 3 * i$	Algorithm loop6(n) $i = 1$ while $i \leq n$ $\quad i = i + i$
Algorithm loop7(n) $i = 1$ while $i \leq n * n$ $\quad i = 2 * i$	Algorithm loop8(n) $i = 1$ while $i \leq n * n$ $\quad i = 3 * i$	Algorithm loop9(n) $i = 1$ $j = n * n$ while $i \leq j$ $\quad i = 2 * i$ $\quad j = j - 1$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n \log n)$	$\Theta(n^3)$	$\Theta(\log n)$	$\Theta(\sqrt{n} \log n)$	$\Theta(n^2)$	$\Theta(\frac{\log n}{\log \log n})$	$\Theta((\log n)^2)$	$\Theta(n)$	
loop1	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.1
loop2	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.2
loop3	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.3
loop4	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.4
loop5	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.5
loop6	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.6
loop7	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.7
loop8	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.8
loop9	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	242.9

Opgave/Problem 243

Algorithm loop1(n) $i = n$ while $i > 0$ if i ulige then $i = i - 1$ else $i = i/2$	Algorithm loop2(n) $i = n$ while $i \leq n * n$ $i = 2 * i$	Algorithm loop3(n) $s = n$ while $s > 0$ $s = \lfloor s/2 \rfloor$
Algorithm loop4(n) $i = 2$ while $i \leq n$ $i = i * i$	Algorithm loop5(n) $s = 2$ while $s \leq n$ $s = s * s$	Algorithm loop6(n) $i = 0$ $s = 0$ $q = 0$ while $q \leq n$ $i = i + 1$ $s = s + i$ $q = q + s$
Algorithm loop7(n) $i = 0$ $s = 0$ while $s \leq n$ $i = i + 1$ $s = s + i$	Algorithm loop8(n) $i = 1$ $j = 1$ $s = 0$ while $s \leq n$ while $j \leq s$ $j = 2 * j$ $s = s + i$ $i = i + 1$	Algorithm loop9(n) $j = n$ $i = 1$ while $j \geq 0$ $j = j - i$ $i = i + 1$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2)$	$\Theta(n \log n)$	$\Theta(\sqrt[3]{n})$	$\Theta(\log \log n)$	$\Theta(n)$	$\Theta((\log n)^2)$	$\Theta(\log n)$	$\Theta(n^3)$	$\Theta(\sqrt{n})$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	☐ I	243.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	☐ I	243.2
loop3	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	☐ I	243.3
loop4	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	☐ I	243.4
loop5	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	☐ H	☐ I	243.5
loop6	☐ A	☐ B	☒ C	☐ D	☐ E	☐ F	☐ G	☐ H	☐ I	243.6
loop7	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒ I	243.7
loop8	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒ I	243.8
loop9	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	☒ I	243.9

Opgave/Problem 244

Algorithm loop1(n) $s = 0$ $i = 1$ while $s \leq n$ $s = s + i$ $i = i + 1$	Algorithm loop2(n) $i = 1$ while $i \leq n$ $j = 1$ $k = 1$ while $k \leq n$ $j = j + 1$ $k = k + j$ $i = 2 * i$	Algorithm loop3(n) for $i = 1$ to n $j = i$ while $j \leq n$ $j = 2 * j$
Algorithm loop4(n) $i = 0$ $j = n$ while $i \leq j$ $i = i + 1$ $j = j - 1$	Algorithm loop5(n) $i = 0$ $j = n$ while $i < j$ $i = i + 2$ $j = j + 1$	Algorithm loop6(n) $i = 1$ $j = 0$ while $i \leq n$ $i = i + i$ while $j < i$ $j = j + 1$
Algorithm loop7(n) $i = 1$ $j = 1$ while $i \leq n$ while $j \leq i$ $j = j + 1$ $i = 2 * i$	Algorithm loop8(n) $i = 1$ $s = 0$ while $s \leq n$ $j = 1$ while $j \leq i$ $j = j + 1$ $s = s + i$ $i = i + 1$	Algorithm loop9(n) $i = 1$ $s = 1$ while $s \leq n * n$ $i = i + 1$ $s = s + i$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta((\log n)^2)$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(\sqrt{n} \log n)$	$\Theta(n)$	$\Theta(n \log n)$	
loop1	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	244.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	☐ H	244.2
loop3	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.3
loop4	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.4
loop5	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.5
loop6	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.6
loop7	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.7
loop8	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.8
loop9	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☒ G	☐ H	244.9

Opgave/*Problem* 245

Algorithm loop1(n) $i = 1$ while $i \leq n$ $j = 0$ while $j \leq i$ $j = j + 1$ $i = 2 * i$	Algorithm loop2(n) $i = 1$ while $i \leq n$ $j = 0$ while $j \leq n$ $j = j + i$ $i = 2 * i$	Algorithm loop3(n) $i = 1$ while $i \leq n$ $j = 1$ while $j \leq i$ $j = j + 1$ $i = 2 * i$
Algorithm loop4(n) $i = n$ while $i > 0$ $i = i - 1$	Algorithm loop5(n) $i = n$ while $i \geq 1$ $j = i$ while $j \leq n$ $j = 2 * j$ $i = i - 1$	Algorithm loop6(n) $s = 0$ $i = 1$ while $i * i \leq n$ for $j = 1$ to i $s = s + 1$ $i = i + 1$
Algorithm loop7(n) $s = 0$ $i = n$ while $i > 1$ for $j = 1$ to i $s = s + 1$ $i = \lfloor i/2 \rfloor$	Algorithm loop8(n) $s = 1$ for $i = 1$ to n $s = s + 1$	Algorithm loop9(n) $s = 1$ for $i = n$ to 1 step -1 $s = s + 1$

Angiv for hver af ovenstående algoritmer udførselstiden som funktion af n i Θ -notation.
Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

[illegible]

Opgave/Problem 246

Algorithm loop1(n) $s = 1$ $i = 1$ while $i \leq n$ for $j = 1$ to i $s = s + 1$ $i = 2 * i$	Algorithm loop2(n) $s = 1$ while $s \leq n$ $s = s + 1$	Algorithm loop3(n) for $i = 1$ to n $j = 0$ while $j \leq n$ $j = j + i$
Algorithm loop4(n) for $i = 1$ to n $j = 1$ while $j \leq i$ $j = 2 * j$	Algorithm loop5(n) for $i = 1$ to n $j = 1$ while $j \leq n$ $j = 2 * j$	Algorithm loop6(n) for $i = 1$ to n $j = i$ while $j > 1$ $j = \lfloor j/2 \rfloor$
Algorithm loop7(n) $i = 0$ while $i \leq n$ $j = i$ while $j > 0$ $j = \lfloor j/2 \rfloor$ $i = i + 1$	Algorithm loop8(n) $i = 1$ $j = 1$ $s = 0$ while $i \leq n$ if $i = j$ then for $k = 1$ to n $s = s + 1$ $j = 2 * j$ $i = i + 1$	Algorithm loop9(n) $i = 1$ $s = 0$ while $i \leq n$ for $j = i$ to n $s = s + 1$ $i = i + i$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2)$	$\Theta(n \log n)$	$\Theta(\log n)$	$\Theta(\frac{\log n}{\log \log n})$	$\Theta(n)$	$\Theta(n^3)$	$\Theta(n\sqrt{n})$	$\Theta((\log n)^2)$	
loop1	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	246.1
loop2	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	☐ H	246.2
loop3	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.3
loop4	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.4
loop5	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.5
loop6	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.6
loop7	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.7
loop8	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.8
loop9	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	246.9

Opgave/Problem 247

Algorithm loop1(n) $i = 1$ while $i \leq n$ $j = 0$ while $j \leq n$ $j = j + 1$ $i = 2 * i$	Algorithm loop2(n) $i = 1$ while $i \leq n$ $j = 1$ while $j \leq i$ $j = 2 * j$ $i = i + 1$	Algorithm loop3(n) $i = 1$ while $i \leq n$ $j = i$ while $j \leq n$ $j = j + 1$ $i = 2 * i$
Algorithm loop4(n) $i = 1$ while $i \leq n$ $j = n$ while $j > 1$ $j = j - 1$ $i = 2 * i$	Algorithm loop5(n) $s = 0$ $i = n$ while $i > 1$ for $j = 1$ to n $s = s + 1$ $i = \lfloor i/2 \rfloor$	Algorithm loop6(n) for $i = 0$ to n $j = 0$ $s = 0$ while $s \leq i$ $j = j + 1$ $s = s + j$
Algorithm loop7(n) for $i = 1$ to n $j = 1$ while $j \leq i$ $j = j + 1$	Algorithm loop8(n) for $i = 1$ to n $j = i$ while $j > 0$ $j = j - 1$	Algorithm loop9(n) $i = 0$ $j = 0$ while $i \leq n$ if $i < j$ then $i = i + 1$ else $j = j + 1$ $i = 0$

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n \log n)$	$\Theta(2^n)$	$\Theta(n^3)$	$\Theta(n^2)$	$\Theta((\log n)^2)$	$\Theta(n)$	$\Theta(\log n)$	$\Theta(n\sqrt{n})$	
loop1	☒	☐	☐	☐	☐	☐	☐	☐	247.1
loop2	☒	☐	☐	☐	☐	☐	☐	☐	247.2
loop3	☒	☐	☐	☐	☐	☐	☐	☐	247.3
loop4	☒	☐	☐	☐	☐	☐	☐	☐	247.4
loop5	☒	☐	☐	☐	☐	☐	☐	☐	247.5
loop6	☐	☐	☐	☐	☐	☐	☐	☒	247.6
loop7	☐	☐	☐	☒	☐	☐	☐	☐	247.7
loop8	☐	☐	☐	☒	☐	☐	☐	☐	247.8
loop9	☐	☐	☐	☒	☐	☐	☐	☐	247.9

Opgave/Problem 248

Algorithm loop1(n)

```

 $i = 1$ 
while  $i \leq n$ 
     $j = 1$ 
    while  $j \leq i$ 
         $j = j + 1$ 
     $i = i + 1$ 

```

Algorithm loop2(n)

```

 $i = 1$ 
while  $i \leq n$ 
     $j = 1$ 
    while  $j \leq n$ 
         $j = j + 1$ 
     $i = i + 1$ 

```

Algorithm loop3(n)

```

 $i = n$ 
 $j = 0$ 
while  $i > 0$ 
    if  $j < i$ 
         $j = j + 1$ 
    else
         $j = 0$ 
         $i = i - 1$ 

```

Algorithm loop4(n)

```

 $s = 0$ 
for  $i = 1$  to  $n$ 
    for  $j = 1$  to  $n$ 
        if  $i = j$  then
            for  $k = 1$  to  $n$ 
                 $s = s + 1$ 

```

Algorithm loop5(n)

```

 $s = 0$ 
 $i = n$ 
while  $i > 0$ 
    for  $j = 1$  to  $i$ 
         $s = s + 1$ 
     $i = i - 1$ 

```

Algorithm loop6(n)

```

 $s = 1$ 
for  $i = 1$  to  $n$ 
    for  $j = 1$  to  $n$ 
         $s = s + 1$ 

```

Algorithm loop7(n)

```

 $s = 1$ 
for  $i = 1$  to  $n$ 
    for  $j = 1$  to  $n$ 
         $s = s + 1$ 
    for  $k = 1$  to  $n$ 
         $s = s + 1$ 

```

Algorithm loop8(n)

```

 $s = 1$ 
for  $i = 1$  to  $n$ 
    for  $j = i$  to  $n$ 
         $s = s + 1$ 

```

Algorithm loop9(n)

```

 $s = 1$ 
for  $i = n$  to  $1$  step  $-1$ 
    for  $j = n$  to  $1$  step  $-1$ 
         $s = s + 1$ 

```

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n \log n)$	$\Theta(n)$	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n^3)$	$\Theta((\log n)^2)$	$\Theta(\sqrt{n} \log n)$	$\Theta(n^2)$	
loop1	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.1
loop2	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.2
loop3	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.3
loop4	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.4
loop5	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.5
loop6	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.6
loop7	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.7
loop8	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.8
loop9	☐ A	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☒	248.9

Opgave/*Problem* 249

Algorithm loop1(n) for $i = 1$ to n for $j = 1$ to i $k = 1$ while $k \leq i + j$ $k = 2 * k$	Algorithm loop2(n) $s = 0$ for $i = 1$ to n for $j = 1$ to $i * i$ $s = s + 1$	Algorithm loop3(n) $s = 0$ for $i = 1$ to n for $j = 1$ to n for $k = 1$ to n $s = s + 1$
Algorithm loop4(n) $s = 0$ for $i = 1$ to n for $j = i$ to n for $k = i$ to j $s = s + 1$	Algorithm loop5(n) $s = 0$ $j = 0$ for $i = 1$ to n $j = j + i$ for $k = 1$ to j $s = s + 1$	Algorithm loop6(n) $s = 1$ for $i = 1$ to n for $j = 1$ to i for $k = j$ to i $s = s + 1$
Algorithm loop7(n) $s = 1$ for $i = 1$ to $n * n$ for $j = 1$ to n $s = s + 1$		

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

Indicate for each of the above algorithms the running time as a function of n in Θ -notation.

	$\Theta(n^2 \cdot \log n)$	$\Theta(n^3)$	$\Theta(n \log n)$	$\Theta(\log n)$	$\Theta(n^2)$	$\Theta((\log n)^2)$	$\Theta(n)$	$\Theta(n\sqrt{n})$	
loop1	☒	☐ B	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.1
loop2	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.2
loop3	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.3
loop4	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.4
loop5	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.5
loop6	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.6
loop7	☐ A	☒	☐ C	☐ D	☐ E	☐ F	☐ G	☐ H	249.7

Amortiseret analyse/*Amortized analysis*

Opgave/*Problem 250* (4 %)

Antag en liste L bruges til at opbevare en mængde af forskellige tal. Følgende to operationer ønskes understøttet: $\text{ADD}(x)$ tilføjer x til listen (x antages at være forskellig fra alle tal i listen), og REMOVELARGERHALF fjerner og returnerer de $\lceil |L|/2 \rceil$ største elementer fra listen. $\text{ADD}(x)$ tilføjer blot det nye element bagerst i listen L i worst-case $O(1)$ tid. REMOVELARGERHALF anvender først deterministisk selektion til at finde det $\lfloor |L|/2 \rfloor + 1$ mindste element e i L i worst-case $O(|L|)$ tid, hvorefter L løbes igennem og alle elementer fjernes der er større end eller lig e .

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at begge operationer tager amortiseret $O(1)$ tid.

Assume a list L is used to store a set of distinct integers. The following two operations are to be supported: $\text{ADD}(x)$ adds x to the list (where x is assumed to be distinct from all integers already in the list), and REMOVELARGERHALF removes and returns the $\lceil |L|/2 \rceil$ largest elements from the list. $\text{ADD}(x)$ simply appends the new element to the end of the list L in worst-case $O(1)$ time. REMOVELARGERHALF first uses deterministic selection to find the $\lfloor |L|/2 \rfloor + 1$ smallest element e in L in worst-case $O(|L|)$ time, after which L is scanned and all elements greater than or equal to e are removed.

For each of the following functions, indicate whether it is a potential function with which one can argue that both operations take amortized $O(1)$ time.

	Ja/Yes	Nej/No	
$ L $	☒	☐	250.1
$\log L $	☐	☒	250.2
$2 L $	☒	☐	250.3
$ L /2$	☒	☐	250.4
$ L ^2$	☐	☒	250.5
$ L \cdot \log L $	☐	☒	250.6

Opgave/Problem 251 (4 %)

Antag at et array X af størrelse n indeholder to stakke S og T , henholdsvis af størrelse s og t , således at $X[1..s] = S$ og $X[n+1-t..n] = T$, hvor toppen af de to stakke er henholdsvis $X[s]$ og $X[n+1-t]$. Når X bliver fuld, dvs. $s+t = n$, allokeres et dobbelt så stort array til X , og S og T kopieres over i dette array.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at stakoperationerne PUSH og POP på de to stakke tager amortiseret $O(1)$ tid.

Assume an array X of size n contains two stacks S and T of sizes s and t , respectively, such that $X[1..s] = S$ and $X[n+1-t..n] = T$, where the tops of the two stacks are $X[s]$ and $X[n+1-t]$, respectively. When X becomes full, i.e., $s+t = n$, a new array of double the size is allocated for X , and S and T are copied into this array.

For each of the following functions, indicate whether it is a potential function with which one can argue that the stack operations PUSH and POP on the two stacks take amortized $O(1)$ time.

	Ja/Yes	Nej/No	
$s+t$	☐ A	☒ B	251.1
$t-s$	☐ A	☒ B	251.2
$s+n-t$	☐ A	☒ B	251.3
$\max\{0, 2(s+t) - n\}$	☒ A	☐ B	251.4
$n-s-t$	☐ A	☒ B	251.5
$\max\{0, 2s+3t-n\}$	☒ A	☐ B	251.6

Opgave/Problem 252 (4 %)

Givet en sorteret liste L med $N = 2^k - 1$ elementer, for et positivt heltal k , så kan man i $O(N)$ tid konstruere et perfekt balanceret binært søgetræ indeholdende L . I det følgende antages at vi kun laver sletninger i søgetræet, som beskrevet i [CLRS, kapitel 12.3], dvs. sletninger forsøger ikke at holde træet balanceret. Lad n betegne det aktuelle antal elementer i træet. For at holde træets højde logaritmisk i n , genopbygges træet som et perfekt balanceret binært søgetræ i $O(n)$ tid når halvdelen af elementerne er blevet slettet, dvs. når $n < N/2$, hvor N betegner antallet af elementer i træet sidste gang det blev genopbygget.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at slettelser tager amortiseret $O(\log n)$ tid.

Given a sorted list L with $N = 2^k - 1$ elements, for a positive integer k , one can in $O(N)$ time construct a perfectly balanced binary search tree containing L . In the following, we assume that we only perform deletions in the search tree, as described in [CLRS, Chapter 12.3], i.e., deletions do not attempt to keep the tree balanced. Let n denote the current number of elements in the tree. To keep the height of the tree logarithmic in n , the tree is reconstructed as a perfectly balanced binary search tree in $O(n)$ time when half of the elements have been deleted, i.e., when $n < N/2$, where N denotes the number of elements in the tree the last time it was reconstructed.

For each of the following functions, indicate whether it is a potential function with which one can argue that deletions take amortized $O(\log n)$ time.

	Ja/Yes	Nej/No	
N	☒	☐	252.1
n	☐	☒	252.2
$N - n$	☒	☐	252.3
$(N - n) \cdot \log n$	☒	☐	252.4
$\log n$	☐	☒	252.5
$(N - n) \cdot \log N$	☒	☐	252.6

Opgave/Problem 253 (4 %)

Betragt en liste $L = (x_1, \dots, x_N)$ af N heltal, hvorpå vi kan udføre følgende to operationer: $\text{APPEND}(x)$ tilføjer heltallet x til sidst i listen, og ADDPAIRS erstatter for alle $i = 1.. \lfloor |L|/2 \rfloor$ det $2i - 1$ 'te og $2i$ 'te tal med deres sum, således at den nye liste har længde $\lceil |L|/2 \rceil$. F.eks. resulterer ADDPAIRS på listen 3, 5, 7, 4, 11, 2, 6 i den nye liste 8, 11, 13, 6. Worst-case tiden for APPEND og ADDPAIRS er henholdsvis $O(1)$ og $O(|L|)$.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at begge operationer tager amortiseret $O(1)$ tid.

Consider a list $L = (x_1, \dots, x_N)$ of N integers, on which we can perform the following two operations: $\text{APPEND}(x)$ appends the integer x to the end of the list, and ADDPAIRS replaces for all $i = 1.. \lfloor |L|/2 \rfloor$ the $2i - 1$ 'th and $2i$ 'th integers with their sum, such that the new list has length $\lceil |L|/2 \rceil$. For example, ADDPAIRS on the list 3, 5, 7, 4, 11, 2, 6 results in the new list 8, 11, 13, 6. The worst-case time for APPEND and ADDPAIRS is $O(1)$ and $O(|L|)$, respectively.

For each of the following functions, indicate whether it is a potential function with which one can argue that both operations take amortized $O(1)$ time.

	Ja/Yes	Nej/No	
N	☒	☐	253.1
$\log N$	☐	☒	253.2
$N + \log N$	☒	☐	253.3
$N/2$	☒	☐	253.4
$N \cdot \log N$	☐	☒	253.5
$\sqrt{N}$	☐	☒	253.6

Opgave/Problem 254 (4 %)

Betragt en uordnet liste af n heltal, hvor vi i $O(1)$ tid kan indsætte et nyt heltal, og i $O(n)$ tid kan udføre operationen `NEGATEREMOVE`, som fjerner alle ikke-positive tal ($x \leq 0$) fra listen og erstatter ethvert positivt heltal $y > 0$ med det tilsvarende negative heltal $-y$. $\text{NEGATEREMOVE}(3, -4, -2, 7, 6, -2) = (-3, -7, -6)$.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at både indsættelser og `NEGATEREMOVE` tager amortiseret $O(1)$ tid, hvor P er antal positive tal i listen og N er antal ikke-positive tal i listen.

Consider an unordered list of n integers, where we can in $O(1)$ time insert a new integer, and in $O(n)$ time perform the operation `NEGATEREMOVE`, which removes all non-positive integers ($x \leq 0$) from the list and replaces each positive integer $y > 0$ with the corresponding negative integer $-y$. $\text{NEGATEREMOVE}(3, -4, -2, 7, 6, -2) = (-3, -7, -6)$.

For each of the following functions, indicate whether it is a potential function with which one can argue that both insertions and `NEGATEREMOVE` take amortized $O(1)$ time, where P is the number of positive integers in the list and N is the number of non-positive integers in the list.

	Ja/Yes	Nej/No	
N	☐ A	☒	254.1
P	☐ A	☒	254.2
$2N + P$	☐ A	☒	254.3
$N + P$	☐ A	☒	254.4
$N + 2P$	☒	☐ B	254.5
$P + N/2$	☒	☐ B	254.6

Opgave/Problem 255 (4 %)

Betragt et rød-sort søgetræ udvidet med operation `INSERTCUT`(x), som indsætter elementet x i søgetræet og sletter alle elementer der er mindre end x fra søgetræet. Hvis indsættelsen sletter k elementer, så tager dette worst-case $O((k+1)\log n)$ tid, hvor n er antal elementer i træet før `INSERTCUT` udføres.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at `INSERTCUT` tager amortiseret $O(\log n)$.

Consider a red-black search tree extended with the operation `INSERTCUT`(x), which inserts the element x into the search tree and deletes all elements less than x from the search tree. If the insertion deletes k elements, this takes worst-case $O((k+1)\log n)$ time, where n is the number of elements in the tree before `INSERTCUT` is performed.

For each of the following functions, indicate whether it is a potential function with which one can argue that `INSERTCUT` takes amortized $O(\log n)$.

	Ja/Yes	Nej/No	
$\log n$	☐ A	☒	255.1
$n \cdot \log n$	☒	☐ B	255.2
n	☐ A	☒	255.3
$\sum_{i=1}^n \log i$	☒	☐ B	255.4
$k \cdot \log n$	☐ A	☒	255.5

Opgave/Problem 256 (4 %)

Antag vi ønsker at gemme en mængde af n tal vha. hashing med linear probing i et array af størrelse N . Vi garanterer at arrayet altid er mellem $1/4$ og $3/4$ fyldt. Hvis der bliver færre end $N/4$ eller flere end $3N/4$ tal i mængden genindsætter vi alle tal i et nyt array af størrelse $2n$, dvs. det nye array er $1/2$ fyldt.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at det totale antal genindsættelser i hashtabellerne er amortiseret $O(1)$ per indsættelse og slettelse i mængden.

Assume we want to store a set of n integers using hashing with linear probing in an array of size N . We guarantee that the array is always between $1/4$ and $3/4$ full. If there are fewer than $N/4$ or more than $3N/4$ integers in the set, we reinsert all integers into a new array of size $2n$, i.e., the new array is $1/2$ full.

For each of the following functions, indicate whether it is a potential function with which one can argue that the total number of reinsertions in the hash tables is amortized $O(1)$ per insertion and deletion in the set.

	Ja/Yes	Nej/No	
$ 2n - N $	☒	☐ B	256.1
$\min(n, N - n)$	☐ A	☒	256.2
$N - n$	☐ A	☒	256.3
$4 N/2 - n $	☒	☐ B	256.4
N	☐ A	☒	256.5
$N/2 - \min(n, N - n)$	☒	☐ B	256.6

Opgave/Problem 257 (4 %)

En binær max-heap understøtter INSERT og HEAP-EXTRACT-MAX på en max-heap med n elementer i worst-case $O(\log n)$ tid.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at INSERT tager amortiseret $O(\log n)$ tid og HEAP-EXTRACT-MAX tager amortiseret $O(1)$ tid.

A binary max-heap supports INSERT and HEAP-EXTRACT-MAX on a max-heap with n elements in worst-case $O(\log n)$ time.

For each of the following functions, indicate whether it is a potential function with which one can argue that INSERT takes amortized $O(\log n)$ time and HEAP-EXTRACT-MAX takes amortized $O(1)$ time.

	Ja/Yes	Nej/No	
$n \cdot \log n$	☒	☐ B	257.1
$\sum_{i=1}^n \log i$	☒	☐ B	257.2
n	☐ A	☒	257.3
$\log n$	☐ A	☒	257.4
$(\log n)^2$	☐ A	☒	257.5

Opgave/Problem 258 (4 %)

Betragt en binær max-heap implementeret i et array. Overløb håndteres ved at allokere et nyt array af dobbelt størrelse og kopiere indholdet af det gamle array til det nye array. Lad den aktuelle størrelse af arrayet være N og antallet af elementer i heapen n .

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at INSERT og HEAP-EXTRACT-MAX kræver amortiseret $O(\log n)$ tid.

Consider a binary max-heap implemented in an array. Overflow is handled by allocating a new array of double size and copying the contents of the old array to the new array. Let the current size of the array be N and the number of elements in the heap be n .

For each of the following functions, indicate whether it is a potential function with which one can argue that INSERT and HEAP-EXTRACT-MAX require amortized $O(\log n)$ time.

	Ja/Yes	Nej/No	
n	☐ A	☒ B	258.1
$ 2n - N $	☒ A	☐ B	258.2
$\max(0, 2n - N)$	☒ A	☐ B	258.3
$N - 2n$	☐ A	☒ B	258.4
$\log n$	☐ A	☒ B	258.5
$\log(N/n)$	☐ A	☒ B	258.6

Opgave/Problem 259 (4 %)

Betragt en stak implementeret i et array, hvor overløb håndteres ved at allokere et nyt større array og kopiere indholdet af det gamle array til det nye array. Angiv den amortiserede tid for en PUSH operation, når det nye array har nedenstående størrelse N og n er antal elementer på stakken før PUSH operationen.

Consider a stack implemented in an array, where overflow is handled by allocating a new larger array and copying the contents of the old array to the new array. Indicate the amortized time for a PUSH operation, when the new array has the following size N and n is the number of elements on the stack before the PUSH operation.

	$O(1)$	$O(\log n)$	$O(\sqrt{n})$	$O(n)$	
$N = n + 1$	☐ A	☐ B	☐ C	☒ D	259.1
$N = n + \lceil \sqrt{n} \rceil$	☐ A	☐ B	☒ C	☐ D	259.2
$N = \lceil \frac{3}{2}n \rceil$	☒ A	☐ B	☐ C	☐ D	259.3
$N = 2n$	☒ A	☐ B	☐ C	☐ D	259.4
$N = 3n$	☒ A	☐ B	☐ C	☐ D	259.5

Opgave/Problem 260 (4 %)

En binær max-heap understøtter MAX-HEAP-INSERT og HEAP-EXTRACT-MAX i worst-case tid $O(\log n)$, hvor n er antal elementer i heapen. Vi vil nu understøtte operationen DELETE, der givet en pointer til et element i heapen, sletter elementet fra heapen. Vi implementerer DELETE ved blot at markere elementet som slettet i worst-case $O(1)$ tid. Når vi udfører HEAP-EXTRACT-MAX gentager vi denne indtil det første ikke-markerede element bliver returneret. Dvs. hvis HEAP-EXTRACT-MAX sletter D markerede elementer bliver worst-case tiden $O((D+1) \log N)$, hvor N er antallet af markerede og ikke-markerede elementer i heapen.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at operationerne MAX-HEAP-INSERT, DELETE, og HEAP-EXTRACT-MAX tager amortiseret $O(\log N)$ tid, hvor M betegner antallet af markerede elementer i heapen.

A binary max-heap supports MAX-HEAP-INSERT and HEAP-EXTRACT-MAX in worst-case time $O(\log n)$, where n is the number of elements in the heap. We now want to support the operation DELETE, which given a pointer to an element in the heap, deletes the element from the heap. We implement DELETE by simply marking the element as deleted in worst-case $O(1)$ time. When we perform HEAP-EXTRACT-MAX, we repeat this until the first unmarked element is returned. I.e., if HEAP-EXTRACT-MAX deletes D marked elements, the worst-case time is $O((D+1) \log N)$, where N is the number of marked and unmarked elements in the heap.

For each of the following functions, indicate whether it is a potential function with which one can argue that the operations MAX-HEAP-INSERT, DELETE, and HEAP-EXTRACT-MAX take amortized $O(\log N)$ time, where M denotes the number of marked elements in the heap.

	Ja/Yes	Nej/No	
N	☐ A	☒ B	260.1
M	☐ A	☒ B	260.2
$N \cdot \log N$	☒ A	☐ B	260.3
$M \cdot \log N$	☒ A	☐ B	260.4
$N \cdot \log M$	☐ A	☒ B	260.5
$M \cdot \log M$	☐ A	☒ B	260.6

Opgave/Problem 261 (4 %)

En binær max-heap understøtter INSERT og HEAP-EXTRACT-MAX på en heap med n elementer i worst-case $O(\log n)$ tid. Bemærk den samme værdi kan være indsat flere gange i en max-heap. Vi ønsker nu at ændre HEAP-EXTRACT-MAX, således at den fjerner alle forekomster af maximum værdien fra heapen, d.v.s. den oprindelige HEAP-EXTRACT-MAX operation gentages indtil roden indeholder en mindre værdi eller at heapen er tom. Hvis maximum forekommer m gange i heapen, så vil HEAP-EXTRACT-MAX operationen tage worst-case $O(m \log n)$ tid.

Angiv for hver af nedenstående funktioner om det er en potentialefunktion, hvormed man kan argumentere for at operationerne INSERT og HEAP-EXTRACT-MAX tager amortiseret $O(\log n)$ tid. Antal elementer i heapen betegnes n og antallet af forskellige elementer i heapen N , hvor $N \leq n$.

A binary max-heap supports INSERT and HEAP-EXTRACT-MAX on a heap with n elements in worst-case $O(\log n)$ time. Note that the same value can be inserted multiple times in a max-heap. We now want to change HEAP-EXTRACT-MAX, such that it removes all occurrences of the maximum value from the heap, i.e., the original HEAP-EXTRACT-MAX operation is repeated until the root contains a smaller value or the heap is empty. If the maximum occurs m times in the heap, then the HEAP-EXTRACT-MAX operation will take worst-case $O(m \log n)$ time.

For each of the following functions, indicate whether it is a potential function with which one can argue that the operations INSERT and HEAP-EXTRACT-MAX take amortized $O(\log n)$ time. The number of elements in the heap is denoted n and the number of distinct elements in the heap N , where $N \leq n$.

	Ja/Yes	Nej/No	
n	☐	☒	261.1
N	☐	☒	261.2
$n - N$	☐	☒	261.3
$n \cdot \log n$	☒	☐	261.4
$N \cdot \log n$	☐	☒	261.5
$(n - N) \cdot \log n$	☒	☐	261.6

Invarianter/*Invariants*

Opgave/*Problem 262* (4 %)

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

Given two non-negative integers n and m , the below algorithm computes $n \cdot m$.

Algorithm Multiplication(n, m)
 Input : $n \geq 0 \wedge m \geq 0$
 Output : $r = n_0 \cdot m_0$
 Method : $r \leftarrow 0$
 { I } **while** $n > 0$ **do**
 if n is odd **then**
 $r \leftarrow r + m$
 $n \leftarrow n - 1$
 else
 $m \leftarrow m * 2$
 $n \leftarrow n/2$

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

For each of the following statements, indicate whether it is an invariant I for the algorithm Multiplication, where n_0 and m_0 denote the initial values of n and m , respectively.

	Ja/Yes	Nej/No	
$0 \leq n \leq n_0$	☒	☐ B	262.1
$0 \leq m \leq m_0$	☐ A	☒	262.2
$r = m \cdot n$	☐ A	☒	262.3
$n_0 \cdot m_0 = r + n \cdot m$	☒	☐ B	262.4
$n_0 \cdot m_0 + r = n \cdot m$	☐ A	☒	262.5

Opgave/Problem 263 (4 %)

Givet positive heltal x og y , så beregner nedenstående algoritme x^y .

Given positive integers x and y , the below algorithm computes x^y .

Algorithm Power(x, y)
 Input : $x \geq 1 \wedge y \geq 1$
 Output : $r = x^y$
 Method : $r \leftarrow 1$
 { I } **while** $y \geq 1$ **do**
 if y is odd **then**
 $y \leftarrow y - 1$
 $r \leftarrow r * x$
 else
 $y \leftarrow y/2$
 $x \leftarrow x * x$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Power, hvor x_0 og y_0 angiver start værdierne for x og y .

For each of the following statements, indicate whether it is an invariant I for the algorithm Power, where x_0 and y_0 denote the initial values of x and y .

	Ja/Yes	Nej/No	
$r = x^y$	☐ A	☒	263.1
$r = x_0^{y_0}$	☐ A	☒	263.2
$r = x_0^{y_0 - y}$	☐ A	☒	263.3
$x_0^{y_0} = r \cdot x^y$	☒	☐ B	263.4
$x^y = r \cdot x_0^{y_0}$	☐ A	☒	263.5

Opgave/Problem 264 (4 %)

Givet et positivt heltal n , så beregner nedenstående algoritme n^2 .

Given a positive integer n , the below algorithm computes n^2 .

```

Algorithm Square( $n$ )
Input   :  $n \geq 1$ 
Output  :  $r = n^2$ 
Method  :  $r \leftarrow 0$ 
           $i \leftarrow 0$ 
          {I} while  $i < n$  do
               $i \leftarrow i + 1$ 
               $r \leftarrow r + i$ 
           $r \leftarrow 2 * r - n$ 
    
```

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Square.

For each of the following statements, indicate whether it is an invariant I for the algorithm Square.

	Ja/Yes	Nej/No	
$i \geq 0 \wedge r = i^2$	☐ A	☒	264.1
$0 \leq i \leq n$	☒	☐ B	264.2
$i \geq 0 \wedge r = i(i+1)/2$	☒	☐ B	264.3
$r \geq i \geq 0$	☒	☐ B	264.4
$0 \leq r \leq n$	☐ A	☒	264.5

Opgave/Problem 265 (4 %)

Givet et array $A[1..n]$ indeholdende $n \geq 1$ heltal og et heltal x , så beregner nedenstående algoritme antallet af forekomster af x i A , dvs. $\text{count}(x, A) = |\{i \mid 1 \leq i \leq n \wedge A[i] = x\}|$.

Given an array $A[1..n]$ containing $n \geq 1$ integers and an integer x , the below algorithm computes the number of occurrences of x in A , i.e., $\text{count}(x, A) = |\{i \mid 1 \leq i \leq n \wedge A[i] = x\}|$.

Algorithm Count(x, A)
 Input : $x \wedge A[1..n] \wedge n \geq 1$
 Output : $r = \text{count}(x, A)$
 Method : $i \leftarrow 0$
 $r \leftarrow 0$
 { I } **while** $i < n$ **do**
 $i \leftarrow i + 1$
 if $x = A[i]$ **then**
 $r \leftarrow r + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for ovenstående algoritme Count.

For each of the following statements, indicate whether it is an invariant I for the above algorithm Count.

	Ja/Yes	Nej/No	
$i \leq n$	☒	☐	265.1
$r = \text{count}(x, A[1..i])$	☒	☐	265.2
$r = \text{count}(x, A[1..i + 1])$	☐	☒	265.3
$i < n$	☐	☒	265.4
$r = 0$	☐	☒	265.5

Opgave/Problem 266 (4 %)

Antag at et sorteret array $A[1..n]$ indeholder $n \geq 2$ forskellige heltal, dvs. $A[1] < A[2] < \dots < A[n-1] < A[n]$. Givet et positivt heltal $x > 0$, identificer nedenstående algoritme om der findes $1 \leq i < j \leq n$, hvor $x = A[j] - A[i]$.

Assume that a sorted array $A[1..n]$ contains $n \geq 2$ different integers, i.e., $A[1] < A[2] < \dots < A[n-1] < A[n]$. Given a positive integer $x > 0$, the below algorithm identifies whether there exist $1 \leq i < j \leq n$ such that $x = A[j] - A[i]$.

Algorithm FindDiff($A[1..n], x$)
 Input : $x > 0 \wedge A[1..n] \wedge n \geq 2 \wedge A[1] < \dots < A[n]$
 Output : $1 \leq i < j \leq n$, where $x = A[j] - A[i]$;
 or $j = n + 1$ if no such pair exists
 Method : $i \leftarrow 1$;
 $j \leftarrow 1$;
 {I} **while** $j \leq n$ **and** $x \neq A[j] - A[i]$ **do**
 if $x > A[j] - A[i]$ **then**
 $j \leftarrow j + 1$
 else
 $i \leftarrow i + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen FindDiff.

For each of the following statements, indicate whether it is an invariant I for the algorithm FindDiff.

	Ja/Yes	Nej/No	
$i \leq j$	☒	☐	266.1
$1 \leq i \leq j \leq n$	☐	☒	266.2
$x \neq A[j] - A[i]$	☐	☒	266.3
$\forall 1 \leq i' < j' \leq n : (i' \geq i \vee j' \geq j) \Rightarrow x \neq A[j'] - A[i']$	☐	☒	266.4
$\forall 1 \leq i' < j' \leq n : (i' < i \vee j' < j) \Rightarrow x \neq A[j'] - A[i']$	☒	☐	266.5

Opgave/Problem 267 (4 %)

Lad $\|x\|$ betegne antal bit med værdien 1 i den binære repræsentation af et ikke-negativt heltal x , f.eks. $\|14_{10}\| = \|1110_2\| = 3$. Nedenstående algoritme beregner $\|x\|$.

Let $\|x\|$ denote the number of bits with value 1 in the binary representation of a non-negative integer x , e.g., $\|14_{10}\| = \|1110_2\| = 3$. The below algorithm computes $\|x\|$.

```

Algorithm Bits( $x$ )
Input   :  $x \geq 0$ 
Output  :  $r = \|x\|$ 
Method  :  $r \leftarrow 0$ ;
          {I} while  $x > 0$  do
              if  $x$  is odd then
                   $x \leftarrow x - 1$ ;
                   $r \leftarrow r + 1$ 
               $x \leftarrow x/2$ 
    
```

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Bits, hvor x_0 angiver den initielle værdi af x .

For each of the following statements, indicate whether it is an invariant I for the algorithm Bits, where x_0 denotes the initial value of x .

	Ja/Yes	Nej/No	
$r = \ x\ $	☐	☒	267.1
$r + \ x\ = \ x_0\ $	☒	☐	267.2
$r + \ x_0\ = \ x\ $	☐	☒	267.3
$r + \ x\ \leq x_0$	☒	☐	267.4
$\ x\ + 2^r = x_0$	☐	☒	267.5

Opgave/Problem 268 (4 %)

Nedenstående algoritme beregner den heltallige 2-tals-logaritme af n , dvs. $\text{intlog}(n) = \lfloor \log_2 n \rfloor$.

The below algorithm computes the integer base-2 logarithm of n , i.e., $\text{intlog}(n) = \lfloor \log_2 n \rfloor$.

Algorithm Log2(n)
 Input : $n \geq 2$
 Output : $r = \text{intlog}(n) = \lfloor \log_2 n \rfloor$
 Method : $i \leftarrow 1$;
 $r \leftarrow 1$;
 $p \leftarrow 2$;
 { I } **while** $2p \leq n$ **do**
 if $p * p \leq n$ **then**
 $p \leftarrow p * p$;
 $r \leftarrow 2 * r$
 else
 $p \leftarrow 2 * p$;
 $r \leftarrow r + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Log2.

For each of the following statements, indicate whether it is an invariant I for the algorithm Log2.

	Ja/Yes	Nej/No	
$1 \leq r < p$	☒	☐	268.1
$2p \leq n$	☐	☒	268.2
$p = 2^r$	☒	☐	268.3
$p = 2r$	☐	☒	268.4
$p = 2^{\text{intlog}(p)}$	☒	☐	268.5

Opgave/Problem 269 (4 %)

Antag at et array $A[1..n-1]$ indeholder $n-1$ forskellige tal fra mængden $\{1, 2, 3, \dots, n\}$, hvor $n \geq 2$. Nedenstående algoritme Missing identificerer det element $r \in \{1, 2, 3, \dots, n\}$ som ikke er i A .

Assume that an array $A[1..n-1]$ contains $n-1$ different numbers from the set $\{1, 2, 3, \dots, n\}$, where $n \geq 2$. The below algorithm Missing identifies the element $r \in \{1, 2, 3, \dots, n\}$ that is not in A .

Algorithm Missing($A[1..n-1]$)

Input : $A[1..n-1] \wedge n \geq 2 \wedge A[i] \in \{1, 2, \dots, n\} \wedge A[i] \neq A[j]$ for $1 \leq i < j < n$

Output : $r \in \{1, 2, 3, \dots, n\} \setminus A$

Method : $i \leftarrow 1$;
 $x \leftarrow 1$;
 $y \leftarrow A[1]$;
{I} while $i < n-1$ **do**
 $i \leftarrow i + 1$;
 $x \leftarrow x + i$;
 $y \leftarrow y + A[i]$
 $r \leftarrow n + x - y$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Missing.

For each of the following statements, indicate whether it is an invariant I for the algorithm Missing.

	Ja/Yes	Nej/No	
$1 \leq i \leq n-1$	☒	☐	269.1
$y = x - i + A[i]$	☐	☒	269.2
$x = i(i+1)/2$	☒	☐	269.3
$y = \sum_{j=1}^i A[j]$	☒	☐	269.4
$i = i + 1$	☐	☒	269.5

Opgave/Problem 270 (4 %)

Antag at et array $A[1..n]$ indeholder $n \geq 1$ heltal. Positionen i i A siges at være *dominerende* hvis $A[i] > A[j]$ for alle $1 \leq j < i$. Lad $\text{dom}(A)$ angive antal dominerende positioner i i A , hvor $1 \leq i \leq n$. Nedenstående algoritme til venstre beregner $\text{dom}(A)$.

Assume that an array $A[1..n]$ contains $n \geq 1$ integers. Position i in A is said to be dominating if $A[i] > A[j]$ for all $1 \leq j < i$. Let $\text{dom}(A)$ denote the number of dominating positions i in A , where $1 \leq i \leq n$. The below algorithm computes $\text{dom}(A)$.

Algorithm Domination(A)
 Input : Array $A[1..n] \wedge n \geq 1$
 Output : $r = \text{dom}(A)$
 Method : $i \leftarrow 1$;
 $x \leftarrow A[1]$;
 $r \leftarrow 1$;
 { I } **while** $i < n$ **do**
 $i \leftarrow i + 1$;
 if $A[i] > x$ **then**
 $x \leftarrow A[i]$;
 $r \leftarrow r + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen.

For each of the following statements, indicate whether it is an invariant I for the algorithm.

	Ja/Yes	Nej/No	
$1 \leq i < n$	☐ A	☒ B	270.1
$x = A[i]$	☐ A	☒ B	270.2
$x \geq A[i]$	☒ A	☐ B	270.3
$r = \text{dom}(A[1..n])$	☐ A	☒ B	270.4
$r = \text{dom}(A[1..i])$	☒ A	☐ B	270.5

Opgave/Problem 271 (4 %)

Antag $A[1..n]$ er et sorteret array med n forskellige positive heltal. Lad $\text{squares}(A)$ angive antal $A[i]$ hvor $A[i]^2$ også forekommer i A . F.eks. er $\text{squares}(1, 3, 4, 7, 9, 16) = 3$, da $1^2 = 1$, $3^2 = 9$ og $4^2 = 16$. Nedenstående algoritme Squares beregner $\text{squares}(A)$.

Assume $A[1..n]$ is a sorted array with n different positive integers. Let $\text{squares}(A)$ denote the number of $A[i]$ such that $A[i]^2$ also occurs in A . For example, $\text{squares}(1, 3, 4, 7, 9, 16) = 3$, since $1^2 = 1$, $3^2 = 9$ and $4^2 = 16$. The below algorithm Squares computes $\text{squares}(A)$.

Algorithm Squares($A[1..n]$)
 Input : Array $A[1..n] \wedge 0 < A[1] < A[2] < \dots < A[n]$
 Output : $r = \text{squares}(A)$
 Method : $i \leftarrow 1$;
 $j \leftarrow 1$;
 $r \leftarrow 0$;
 { I } **while** $j \leq n$ **do**
 if $A[i]^2 < A[j]$ **then**
 $i \leftarrow i + 1$
 else if $A[i]^2 = A[j]$ **then**
 $i \leftarrow i + 1$; $j \leftarrow j + 1$; $r \leftarrow r + 1$
 else if $A[i]^2 > A[j]$ **then**
 $j \leftarrow j + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for ovenstående algoritme Squares. Det antages at $A[0] = 0$ og $A[n+1] = +\infty$.

For each of the following statements, indicate whether it is an invariant I for the above algorithm Squares. It is assumed that $A[0] = 0$ and $A[n+1] = +\infty$.

	Ja/Yes	Nej/No	
$1 \leq i \leq j \leq n$	☐ A	☒ B	271.1
$i \leq j$	☒ A	☐ B	271.2
$A[i-1]^2 < A[j] \wedge r = \text{squares}(A[1..j])$	☐ A	☒ B	271.3
$A[i-1]^2 < A[j] \wedge r = \text{squares}(A[1..j-1])$	☒ A	☐ B	271.4
$r = j - 1$	☐ A	☒ B	271.5

Opgave/Problem 272 (4 %)

Givet et positivt heltal n , så beregner nedenstående algoritme n^3 .

Given a positive integer n , the below algorithm computes n^3 .

Algorithm Power3(n)

Input : $n \geq 1$

Output : $r = n^3$

Method : $i \leftarrow 1$

$s \leftarrow 1$

$r \leftarrow 1$

{I} while $i < n$ **do**

$i \leftarrow i + 1$

$s \leftarrow s + 2i - 1$

$r \leftarrow r + 3s - 3i + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Power3.

For each of the following statements, indicate whether it is an invariant I for the algorithm Power3.

	Ja/Yes	Nej/No	
$1 \leq i < n$	☐ A	☒ B	272.1
$1 \leq i \leq s \leq r$	☒ A	☐ B	272.2
$r = i^3$	☒ A	☐ B	272.3
$r = n^3$	☐ A	☒ B	272.4
$s = s + 2i - 1$	☐ A	☒ B	272.5

Opgave/Problem 273 (4 %)

Givet et ikke-negativt heltal x og et positive heltal y , så beregner nedenstående algoritme $\lfloor x/y \rfloor$.

Given a non-negative integer x and a positive integer y , the below algorithm computes $\lfloor x/y \rfloor$.

Algorithm Division(x, y)
 Input : $x \geq 0 \wedge y \geq 1$
 Output : $r = \lfloor x/y \rfloor$
 Method : $r \leftarrow 0$
 $\{I\}$ **while** $x \geq y$ **do**
 $x \leftarrow x - y$
 $r \leftarrow r + 1$

For hvert af følgende udsagn, angiv om det er en invariant I for algoritmen Division, hvor x_0 og y_0 angiver værdierne for henholdsvis x og y i starten.

For each of the following statements, indicate whether it is an invariant I for the algorithm Division, where x_0 and y_0 denote the initial values of x and y , respectively.

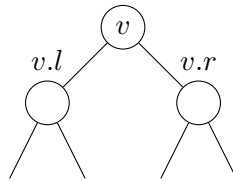
	Ja/Yes	Nej/No	
$r = \lfloor x/y \rfloor$	☐ A	☒ B	273.1
$r = \lfloor x_0/y_0 \rfloor$	☐ A	☒ B	273.2
$r = \lfloor (x_0 - x)/y \rfloor$	☒ A	☐ B	273.3
$x + ry = x_0$	☒ A	☐ B	273.4
$r(x - x_0) = y$	☐ A	☒ B	273.5

Udvidede søgetræer / *Augmented search trees*

Opgave / *Problem 274* (4 %)

Betragt et rød-sort træ hvor hver knude gemmer et par af heltal (*element*, *vægt*), og parrene er sorteret fra venstre-mod-højre efter stigende *element* værdi. For en knude v i træet lader vi $v.e$ og $v.w$ betegne parret (e, w) gemt i knuden. Desuden gemmer v værdien $v.W$ som er summen af vægtene i alle knuder i v 's undertræ, og $v.prefix$ som er den maksimale sum af vægtene et *præfiks* af parrene i v 's undertræ kan have (når parrene sorteres efter element værdi). Angiv hvorledes $v.prefix$ kan beregnes når den tilsvarende information er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Consider a red-black tree where each node stores a pair of integers (*element*, *weight*), and the pairs are sorted from left-to-right according to increasing *element* value. For a node v in the tree we let $v.e$ and $v.w$ denote the pair (e, w) stored in the node. Furthermore, v stores the value $v.W$ which is the sum of the weights in all nodes in v 's subtree, and $v.prefix$ which is the maximum sum of the weights a prefix of the pairs in v 's subtree can have (when the pairs are sorted according to element value). Indicate how $v.prefix$ can be computed when the corresponding information is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.prefix = \max(v.l.prefix, v.l.W + v.w, v.l.prefix + v.w + v.r.prefix) \quad \boxed{A}$$

$$v.prefix = \max(v.l.prefix, v.l.W + v.w, v.l.W + v.w + v.r.prefix) \quad \boxed{X}$$

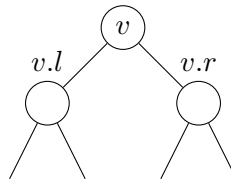
$$v.prefix = \max(v.l.prefix, v.W + v.r.prefix) \quad \boxed{C}$$

$$v.prefix = \max(v.l.W, v.l.W + v.w, v.l.W + v.w + v.r.W) \quad \boxed{D}$$

Opgave/Problem 275 (4 %)

Et interval træ er et rød-sort træ hvor hver knude gemmer præcis ét interval $(low, high)$, og intervallerne er sorteret fra venstre-mod-højre efter stigende low værdi. For en knude v i træet ladet vi $v.low$ og $v.high$ betegne endepunkterne på intervallet gemt i knuden. Desuden gemmer v værdien $v.max$ som er den maximale værdi i et interval gemt i v 's undertræ. Angiv hvorledes $v.max$ kan beregnes når den tilsvarende information er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

An interval tree is a red-black tree where each node stores exactly one interval $(low, high)$, and the intervals are sorted from left-to-right according to increasing low value. For a node v in the tree we let $v.low$ and $v.high$ denote the endpoints of the interval stored in the node. Furthermore, v stores the value $v.max$ which is the maximum value in an interval stored in v 's subtree. Indicate how $v.max$ can be computed when the corresponding information is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$v.max = v.r.max$ ☐ A

$v.max = \max(v.r.max, v.high)$ ☐ B

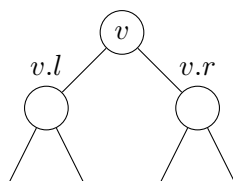
$v.max = \max(v.r.max, v.l.max, v.high)$ ☒ C

$v.max = \max(v.l.high, v.high, v.r.high)$ ☐ D

Opgave/Problem 276 (4 %)

Betragt et søgetræ hvor hver knude v gemmer et tal $v.x$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i en knude v to værdier $v.size$ og $v.avg$, som er henholdsvis antallet af elementer i v 's undertræ og gennemsnittet af alle tallene i v 's undertræ. Angiv hvorledes $v.size$ og $v.avg$ kan beregnes når den tilsvarende information er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Consider a search tree where each node v stores a number $v.x$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, a node v stores two values $v.size$ and $v.avg$, which are respectively the number of elements in v 's subtree and the average of all numbers in v 's subtree. Indicate how $v.size$ and $v.avg$ can be computed when the corresponding information is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.size = v.size + v.l.size + v.r.size \quad \boxed{A}$$

$$v.size = 1 + v.l.size + v.r.size \quad \boxed{\times}$$

$$v.size = v.l.size + v.r.size \quad \boxed{C}$$

$$v.size = 1 + (v.r.size - v.l.size) \quad \boxed{D}$$

$$v.avg = (v.x + v.l.avg + v.r.avg)/3 \quad \boxed{A}$$

$$v.avg = (v.x + v.l.avg * v.l.size + v.r.avg * v.r.size)/v.size \quad \boxed{\times}$$

$$v.avg = (v.x + v.l.avg + v.r.avg)/v.size \quad \boxed{C}$$

$$v.avg = v.x + (v.l.avg + v.r.avg)/2 \quad \boxed{D}$$

Opgave/Problem 277 (4 %)

Betragt en liste af n punkter $(x_1, y_1), \dots, (x_n, y_n)$, hvor x_i og y_i er reelle tal og $x_1 < x_2 < \dots < x_n$. Vi ønsker at finde en sammenhængende række af punkter

$$(x_i, y_i), (x_{i+1}, y_{i+1}), \dots, (x_{j-1}, y_{j-1}), (x_j, y_j),$$

hvor $i \leq j$, således at $Y_{i,j} = \sum_{k=i}^j y_k = y_i + y_{i+1} + \dots + y_{j-1} + y_j$ er størst mulig. Vi ønsker for en dynamisk liste af punkter at vedligeholde denne maksimale sammenhængende y -sum, $maxysum$.

Betragt et søgetræ hvor hver knude v gemmer et punkt $(v.x, v.y)$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i en knude v , der i undertræet indeholder punkterne $(x_k, y_k), \dots, (x_\ell, y_\ell)$ fire værdier $v.sum = Y_{k,\ell}$, $v.maxysum = \max_{k \leq i \leq j \leq \ell} Y_{i,j}$, $v.pre = \max\{0, \max_{k \leq j \leq \ell} Y_{k,j}\}$, og $v.suf = \max\{0, \max_{k \leq i \leq \ell} Y_{i,\ell}\}$.

Angiv hvorledes $v.sum$ og $v.maxysum$ kan beregnes når den tilsvarende information (incl. pre og suf) er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

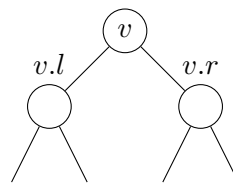
Consider a list of n points $(x_1, y_1), \dots, (x_n, y_n)$, where x_i and y_i are real numbers and $x_1 < x_2 < \dots < x_n$. We want to find a contiguous sequence of points

$$(x_i, y_i), (x_{i+1}, y_{i+1}), \dots, (x_{j-1}, y_{j-1}), (x_j, y_j),$$

where $i \leq j$, such that $Y_{i,j} = \sum_{k=i}^j y_k = y_i + y_{i+1} + \dots + y_{j-1} + y_j$ is as large as possible. We want for a dynamic list of points to maintain this maximum contiguous y -sum, $maxysum$.

Consider a search tree where each node v stores a point $(v.x, v.y)$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, a node v , which in its subtree contains the points $(x_k, y_k), \dots, (x_\ell, y_\ell)$, stores four values $v.sum = Y_{k,\ell}$, $v.maxysum = \max_{k \leq i \leq j \leq \ell} Y_{i,j}$, $v.pre = \max\{0, \max_{k \leq j \leq \ell} Y_{k,j}\}$, and $v.suf = \max\{0, \max_{k \leq i \leq \ell} Y_{i,\ell}\}$.

Indicate how $v.sum$ and $v.maxysum$ can be computed when the corresponding information (incl. pre and suf) is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.sum = v.l.sum + v.r.sum \quad \boxed{A}$$

$$v.sum = v.l.sum + v.y + v.r.sum \quad \boxed{X}$$

$$v.sum = v.l.suf + v.r.pre \quad \boxed{C}$$

$$v.sum = v.l.suf + v.y + v.r.pre \quad \boxed{D}$$

$$v.maxysum = \max\{v.l.maxysum, v.r.maxysum\} \quad \boxed{A}$$

$$v.maxysum = \max\{v.l.maxysum, v.y, v.r.maxysum\} \quad \boxed{B}$$

$$v.maxysum = \max\{v.l.maxysum, v.l.suf + v.y + v.r.pre, v.r.maxysum\} \quad \boxed{X}$$

$$v.maxysum = \max\{v.l.maxysum, v.l.suf + v.r.pre, v.r.maxysum\} \quad \boxed{D}$$

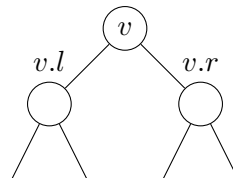
Opgave/Problem 278 (4 %)

Betragt en liste af n punkter $(x_1, y_1), \dots, (x_n, y_n)$, hvor x_i og y_i er reelle tal og $x_1 < x_2 < \dots < x_n$. Et punkt (x_i, y_i) er domineret af et punkt (x_j, y_j) hvis $x_i \leq x_j$ og $y_i \leq y_j$. En liste af punkter er *dominans-fri* hvis og kun hvis intet punkt i listen er domineret af et andet punkt. F.eks. er listen $(3, 7), (5, 3), (7, 6), (13, 4)$ *ikke* dominans-fri, da $(5, 3)$ er domineret af $(13, 4)$. Vi ønsker for en dynamisk liste af punkter at vedligeholde udsagnet om listen er dominans-fri.

Betragt et søgetræ hvor hver knude v gemmer et punkt $(v.x, v.y)$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i v også $v.DF$ og $v.miny$ og $v.maxy$, hvor $v.DF$ er sand hvis og kun hvis mængden af punkterne i v 's undertræ er dominans-fri, og $v.miny$ og $v.maxy$ er den mindste og største y værdi gemt i v 's undertræ. Angiv hvorledes $v.miny$ og $v.DF$ kan beregnes når den tilsvarende information (incl. $maxy$) er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Consider a list of n points $(x_1, y_1), \dots, (x_n, y_n)$, where x_i and y_i are real numbers and $x_1 < x_2 < \dots < x_n$. A point (x_i, y_i) is dominated by a point (x_j, y_j) if $x_i \leq x_j$ and $y_i \leq y_j$. A list of points is dominance-free if and only if no point in the list is dominated by another point. For example, the list $(3, 7), (5, 3), (7, 6), (13, 4)$ is not dominance-free, since $(5, 3)$ is dominated by $(13, 4)$. We want for a dynamic list of points to maintain the statement of whether the list is dominance-free.

Consider a search tree where each node v stores a point $(v.x, v.y)$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, v also stores $v.DF$ and $v.miny$ and $v.maxy$, where $v.DF$ is true if and only if the set of points in v 's subtree is dominance-free, and $v.miny$ and $v.maxy$ are the smallest and largest y value stored in v 's subtree. Indicate how $v.miny$ and $v.DF$ can be computed when the corresponding information (incl. $maxy$) is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.miny = v.l.miny \quad \boxed{A}$$

$$v.miny = \min\{v.l.miny, v.r.miny\} \quad \boxed{B}$$

$$v.miny = \min\{v.l.miny, v.y, v.r.miny\} \quad \boxed{\otimes}$$

$$v.DF = v.l.DF \wedge v.r.DF \quad \boxed{A}$$

$$v.DF = v.l.DF \wedge v.r.DF \wedge v.y < v.l.miny \wedge v.y > v.r.maxy \quad \boxed{\otimes}$$

$$v.DF = v.l.DF \wedge v.r.DF \wedge v.y > v.l.miny \wedge v.y < v.r.maxy \quad \boxed{C}$$

$$v.DF = v.l.DF \wedge v.r.DF \wedge v.y < v.l.miny \quad \boxed{D}$$

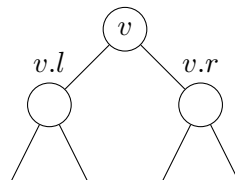
Opgave/Problem 279 (4 %)

Betragt en liste af n par $(x_1, c_1), \dots, (x_n, c_n)$, hvor $x_1 < x_2 < \dots < x_n$ er reelle tal og hvert $c_i \in \{\text{Red, Green, Blue}\}$ er en farve.

Betragt et søgetræ hvor hver knude v gemmer et par $(v.x, v.c)$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i en knude v , en sandhendsværdi $v.mono$, der angiver om alle elementer i v 's undertræ har samme farve, og mængden $v.missing$ af de farver der *ikke* forekommer i v 's undertræ. Angiv hvorledes $v.mono$ og $v.missing$ kan beregnes når den tilsvarende information er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Consider a list of n pairs $(x_1, c_1), \dots, (x_n, c_n)$, where $x_1 < x_2 < \dots < x_n$ are real numbers and each $c_i \in \{\text{Red, Green, Blue}\}$ is a color.

Consider a search tree where each node v stores a pair $(v.x, v.c)$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, a node v stores a boolean value $v.mono$, which indicates whether all elements in v 's subtree have the same color, and the set $v.missing$ of the colors that do not occur in v 's subtree. Indicate how $v.mono$ and $v.missing$ can be computed when the corresponding information is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.mono = v.l.mono \wedge v.r.mono \quad \boxed{\text{A}}$$

$$v.mono = v.l.mono \wedge v.r.mono \wedge v.l.c = v.r.c \wedge v.c = v.r.c \quad \boxed{\times}$$

$$v.mono = v.l.mono \wedge v.r.mono \wedge v.l.c = v.r.c \quad \boxed{\text{C}}$$

$$v.mono = v.l.mono \wedge v.r.mono \wedge v.mono \quad \boxed{\text{D}}$$

$$v.missing = v.l.missing \cup v.r.missing \quad \boxed{\text{A}}$$

$$v.missing = v.l.missing \cap v.r.missing \quad \boxed{\text{B}}$$

$$v.missing = \{v.c\} \cup v.l.missing \cup v.r.missing \quad \boxed{\text{C}}$$

$$v.missing = (v.l.missing \cap v.r.missing) \setminus \{v.c\} \quad \boxed{\times}$$

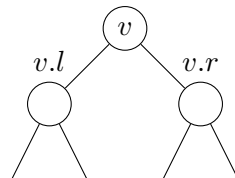
Opgave/Problem 280 (4 %)

Givet en streng T indeholdende bogstaver og start- og slut-parenteser (og), antages at alle positionerne med parenteser er gemt i et søgetræ, sorteret fra venstre-mod-højre efter stigende position. En knude v gemmer en position $v.p$ og den tilhørende parentes $v.c = T[v.p]$ fra T . For $T = \text{"a)b(cd(x)dc(a)}$ gemmes i træet $\langle v.p, v.c \rangle$ parrene $\langle 2, \rangle$, $\langle 4, (\rangle$, $\langle 7, (\rangle$, $\langle 9, \rangle$ og $\langle 12, (\rangle$.

Vi ønsker at vedligeholde information om parenteserne er balancerede. I ovenstående eksempel er parenteserne ") (() (" ikke balancerede, da kun de markerede parenteser går ud mod hinanden. De resterende parenteser ") ((" vil altid være R)-parenteser efterfulgt af L (-parenteser, hvor $R \geq 0$ og $L \geq 0$. I eksemplet har vi $R = 1$ og $L = 2$. I en knude v i træet gemmes disse værdier $v.R$ og $v.L$ for delsekvensen af parenteserne i v 's undertræ. Angiv hvorledes $v.R$ kan beregnes når $v.c =)$ og R og L værdierne er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Given a string T containing letters and opening and closing parentheses (and), we assume that all positions with parentheses are stored in a search tree, sorted from left-to-right according to increasing position. A node v stores a position $v.p$ and the corresponding parenthesis $v.c = T[v.p]$ from T . For $T = \text{"a)b(cd(x)dc(a)}$ the $\langle v.p, v.c \rangle$ pairs $\langle 2, \rangle$, $\langle 4, (\rangle$, $\langle 7, (\rangle$, $\langle 9, \rangle$ and $\langle 12, (\rangle$ are stored in the tree.

We want to maintain information about whether the parentheses are balanced. In the above example the parentheses ") (() (" are not balanced, since only the marked parentheses match each other. The remaining parentheses ") ((" will always be R)-parentheses followed by L (-parentheses, where $R \geq 0$ and $L \geq 0$. In the example we have $R = 1$ and $L = 2$. In a node v in the tree these values $v.R$ and $v.L$ are stored for the subsequence of parentheses in v 's subtree. Indicate how $v.R$ can be computed when $v.c =)$ and the R and L values are known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$$v.R = v.l.R + 1 + v.r.R \quad \boxed{\text{A}}$$

$$v.R = v.l.R + v.r.R + 1 - v.l.L \quad \boxed{\text{B}}$$

$$v.R = v.l.R + \max\{0, v.r.R + 1 - v.l.L\} \quad \boxed{\otimes}$$

$$v.R = v.l.R + 1 + \max\{0, v.r.R - v.l.L\} \quad \boxed{\text{D}}$$

Opgave/Problem 281 (4 %)

For n tal $x_1, \dots, x_n$ ønsker vi at beregne koefficienterne a og b til polynomiet

$$P(y) = \sum_{i=1}^n (x_i - y)^2 = n \cdot y^2 + a \cdot y + b .$$

F.eks. for $x_1 = 2$, $x_2 = 3$, og $x_3 = 5$, har vi polynomiet $P(y) = (2-y)^2 + (3-y)^2 + (5-y)^2 = 3y^2 - 20y + 38$, dvs. $a = -20$ og $b = 38$.

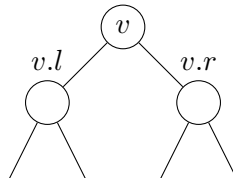
Betragt et søgetræ hvor hver knude v gemmer et tal $v.x$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i en knude v to værdier $v.a$ og $v.b$, som er a og b koefficienterne for $P(y)$ polynomiet defineret ved alle x værdierne i v 's undertræ. Angiv hvorledes $v.a$ og $v.b$ kan beregnes når den tilsvarende information er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

For n numbers $x_1, \dots, x_n$ we want to compute the coefficients a and b for the polynomial

$$P(y) = \sum_{i=1}^n (x_i - y)^2 = n \cdot y^2 + a \cdot y + b .$$

For example, for $x_1 = 2$, $x_2 = 3$, and $x_3 = 5$, we have the polynomial $P(y) = (2 - y)^2 + (3 - y)^2 + (5 - y)^2 = 3y^2 - 20y + 38$, i.e., $a = -20$ and $b = 38$.

Consider a search tree where each node v stores a number $v.x$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, a node v stores two values $v.a$ and $v.b$, which are the a and b coefficients for the $P(y)$ polynomial defined by all x values in v 's subtree. Indicate how $v.a$ and $v.b$ can be computed when the corresponding information is known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).



$v.a = v.l.a + v.r.a + v.x$ ☐ A

$v.a = v.l.a + v.r.a - 2 * v.x$ ☒ B

$v.a = v.l.a * v.r.a * (v.x)^2$ ☐ C

$v.a = v.l.a + v.r.a + (v.x)^2$ ☐ D

$v.b = v.l.b + v.r.b + v.x$ ☐ A

$v.b = v.l.b + v.r.b + (v.x)^2$ ☒ B

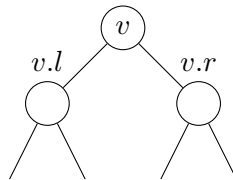
$v.b = v.l.b + v.r.b - 2 * v.x$ ☐ C

$v.b = v.l.b + v.r.b + 2 * v.x$ ☐ D

Opgave/Problem 282 (4 %)

Betragt et søgetræ hvor hver knude v gemmer et tal $v.x$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Derudover gemmes i en knude v tre værdier $v.min$, $v.max$, og $v.closest$. Værdierne $v.min$ og $v.max$ er henholdsvis det mindst og største tal i v 's undertræ, og $v.closest$ den mindste difference mellem to tal i v 's undertræ. Hvis v 's undertræ kun indeholder et tal er $v.closest = +\infty$. Angiv hvorledes $v.closest$ og kan beregnes når min , max og $closest$ værdierne er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

Consider a search tree where each node v stores a number $v.x$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, a node v stores three values $v.min$, $v.max$, and $v.closest$. The values $v.min$ and $v.max$ are respectively the smallest and largest number in v 's subtree, and $v.closest$ is the smallest difference between two numbers in v 's subtree. If v 's subtree contains only one number then $v.closest = +\infty$. Indicate how $v.closest$ can be computed when the min , max , and $closest$ values are known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).

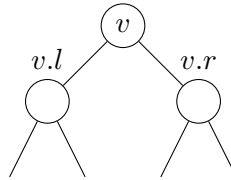


- $v.closest = \min(v.l.closest, v.r.x - v.l.x, v.r.closest)$ ☐ A
- $v.closest = \min(v.l.closest, v.x - v.l.min, v.r.max - v.x, v.r.closest)$ ☐ B
- $v.closest = \min(v.l.closest, v.x - v.l.max, v.r.min - v.x, v.r.closest)$ ☒ C
- $v.closest = \min(v.l.closest, v.r.min - v.l.max, v.r.closest)$ ☐ D
- $v.closest = v.r.closest - v.l.closest$ ☐ E

Opgave/Problem 283 (4 %)

For en sorteret liste af tal $x_1 \leq x_2 \leq \dots \leq x_n$ definerer vi *sum of square gaps* (*ssg*) som $\sum_{i=2..n} (x_i - x_{i-1})^2$. Betragt et rød-sort træ hvor hver knude v gemmer et heltal $v.x$, og knuderne er ordnet venstre-mod-højre efter stigende $v.x$. Desuden gemmer v værdierne $v.min$, $v.max$ og $v.ssg$, som er hhv. det mindste, største og *sum of square gaps* af elementerne i undertræet rodet i v . Angiv hvorledes $v.ssg$ kan beregnes når $v.min$, $v.max$ og $v.ssg$ er kendt ved de to børn $v.l$ og $v.r$ (det kan antages at disse begge eksisterer).

*For a sorted list of numbers $x_1 \leq x_2 \leq \dots \leq x_n$ we define the sum of square gaps (*ssg*) as $\sum_{i=2..n} (x_i - x_{i-1})^2$. Consider a red-black tree where each node v stores an integer $v.x$, and the nodes are ordered left-to-right according to increasing $v.x$. Furthermore, v stores the values $v.min$, $v.max$, and $v.ssg$, which are respectively the smallest, largest, and sum of square gaps of the elements in the subtree rooted at v . Indicate how $v.ssg$ can be computed when $v.min$, $v.max$, and $v.ssg$ are known for the two children $v.l$ and $v.r$ (it can be assumed that both exist).*



- $v.ssg = v.l.ssg + v.r.ssg$ ☐ A
- $v.ssg = v.l.ssg + (v.r.min - v.l.max)^2 + v.r.ssg$ ☐ B
- $v.ssg = v.l.ssg + (v.x - v.l.max)^2 + (v.x - v.r.min)^2 + v.r.ssg$ ☒ C
- $v.ssg = v.l.ssg + (v.r.x - v.l.x)^2 + v.r.ssg$ ☐ D
- $v.ssg = v.l.ssg + (v.x - v.l.x)^2 + (v.x - v.r.x)^2 + v.r.ssg$ ☐ E

Diverse spørgsmål/*Miscellaneous questions*

Opgave/*Problem 284* (4 %)

Angiv for hver af følgende algoritmer best-case, worst-case og forventet køretid på input af størrelse n , hvor input kan indeholde identiske elementer.

For each of the following algorithms, give the best-case, worst-case, and expected running time on input of size n , where the input may contain identical elements.

	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(2^n)$	
Worst-case tid for INSERTION-SORT	☐ A	☐ B	☐ C	☒ D	☐ E	284.1
<i>Worst-case time for INSERTION-SORT</i>						
Worst-case tid for MERGE-SORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.2
<i>Worst-case time for MERGE-SORT</i>						
Worst-case tid for HEAPSORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.3
<i>Worst-case time for HEAPSORT</i>						
Worst-case tid for QUICKSORT	☐ A	☐ B	☐ C	☒ D	☐ E	284.4
<i>Worst-case time for QUICKSORT</i>						
Best-case tid for INSERTION-SORT	☐ A	☒ B	☐ C	☐ D	☐ E	284.5
<i>Best-case time for INSERTION-SORT</i>						
Best-case tid for MERGE-SORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.6
<i>Best-case time for MERGE-SORT</i>						
Best-case tid for HEAPSORT	☐ A	☒ B	☐ C	☐ D	☐ E	284.7
<i>Best-case time for HEAPSORT</i>						
Best-case tid for QUICKSORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.8
<i>Best-case time for QUICKSORT</i>						
Forventet tid for INSERTION-SORT	☐ A	☐ B	☐ C	☒ D	☐ E	284.9
<i>Expected time for INSERTION-SORT</i>						
Forventet tid for MERGE-SORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.10
<i>Expected time for MERGE-SORT</i>						
Forventet tid for HEAPSORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.11
<i>Expected time for HEAPSORT</i>						
Forventet tid for QUICKSORT	☐ A	☐ B	☒ C	☐ D	☐ E	284.12
<i>Expected time for QUICKSORT</i>						

Opgave/*Problem 285* (4 %)

Angiv worst-case tiden for HEAPSORT på et array med n identiske elementer.

Give the worst-case time for HEAPSORT on an array of n identical elements.

$\Theta(\sqrt{n})$ $\Theta(n)$ $\Theta(n \log n)$ $\Theta(n\sqrt{n})$ $\Theta(n^2)$

☐ A ☒ B ☐ C ☐ D ☐ E

Opgave/Problem 286 (4 %)

Angiv hvor mange gange det *største element* i et array med n elementer i worst-case kan blive sammenlignet med andre elementer under udførslen af MERGE-SORT.

Give the number of times the largest element in an array of n elements can be compared with other elements in the worst case during the execution of MERGE-SORT.

$$\Theta(1) \quad \Theta(\log n) \quad \Theta(\sqrt{n}) \quad \Theta(n) \quad \Theta(n \log n)$$

☐ A ☐ B ☐ C ☒ D ☐ E

Opgave/Problem 287 (4 %)

Givet et sorteret array $A[1..n]$ ($A[1] < A[2] < \dots < A[n]$) og et element x , så ønsker vi at finde indekset ℓ således at $A[\ell] \leq x < A[\ell + 1]$. Det antages at $A[1] \leq x < A[n]$. Hvilken af følgende algoritmer er korrekt (kun linierne 2 og 4 varierer i algoritmerne).

Given a sorted array $A[1..n]$ ($A[1] < A[2] < \dots < A[n]$) and an element x , we want to find the index ℓ such that $A[\ell] \leq x < A[\ell + 1]$. It is assumed that $A[1] \leq x < A[n]$. Which of the following algorithms is correct (only lines 2 and 4 vary in the algorithms).

$\ell = 1, h = n + 1$	$\ell = 1, h = n + 1$	$\ell = 1, h = n + 1$	$\ell = 1, h = n + 1$
while $\ell < h$	while $\ell + 1 < h$	while $\ell < h$	while $\ell + 1 < h$
$m = \lfloor (h + \ell)/2 \rfloor$	$m = \lfloor (h + \ell)/2 \rfloor$	$m = \lfloor (h + \ell)/2 \rfloor$	$m = \lfloor (h + \ell)/2 \rfloor$
if $A[m] > x$	if $A[m] > x$	if $A[m] \leq x$	if $A[m] \leq x$
$\ell = m$	$\ell = m$	$\ell = m$	$\ell = m$
else	else	else	else
$h = m$	$h = m$	$h = m$	$h = m$

☐ A ☐ B ☐ C ☒ D

Opgave/Problem 288 (4 %)

Strassens algoritme til multiplikation af kvadratiske $n \times n$ matricer er en del-og-kombiner algoritme. Angiv hvilken rekursionsligning der beskriver udførselstiden af Strassen's algoritme

Strassen's algorithm for multiplication of square $n \times n$ matrices is a divide-and-conquer algorithm. Indicate which recurrence relation describes the execution time of Strassen's algorithm.

$$T(n) \leq 7 \cdot T(n/2) + c \cdot n \quad \text{A}$$

$$T(n) \leq 7 \cdot T(n/4) + c \cdot n^2 \quad \text{B}$$

$$T(n) \leq 7 \cdot T(n/2) + c \cdot n^2 \quad \text{C}$$

$$T(n) \leq 7 \cdot T(n/2) + c \cdot n^3 \quad \text{D}$$

Opgave/Problem 289 (4 %)

Angiv worst-case udførselstiden for hver af følgende algoritmer når input er et array af størrelse n .

Give the worst-case execution time for each of the following algorithms when the input is an array of size n .

	$\Theta(1)$	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^3)$	
BINARY-SEARCH	☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G	289.1
INSERTION-SORT	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	289.2
MERGE-SORT	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	289.3
HEAPSORT	☐ A	☐ B	☐ C	☐ D	☒ E	☐ F	☐ G	289.4
QUICKSORT	☐ A	☐ B	☐ C	☐ D	☐ E	☒ F	☐ G	289.5
PARTITION	☐ A	☐ B	☐ C	☒ D	☐ E	☐ F	☐ G	289.6

Opgave/Problem 290 (4 %)

Givet et array af størrelse n indeholdende tallene $1, 2, \dots, n$ i voksende rækkefølge, hvad er worst-case tiden for følgende sorteringsalgoritmer, når de anvendes på arrayet?

Given an array of size n containing the numbers $1, 2, \dots, n$ in ascending order, what is the worst-case time for the following sorting algorithms when applied to the array?

	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	
INSERTION-SORT	☒ A	☐ B	☐ C	290.1
HEAPSORT	☐ A	☒ B	☐ C	290.2
MERGE-SORT	☐ A	☒ B	☐ C	290.3
QUICKSORT	☐ A	☐ B	☒ C	290.4

Opgave/Problem 291 (4 %)

Selektionsalgoritmen til at finde det *ite* mindste element i et ikke-sorteret list i worst-case tid $O(n)$ [CLRS, kapitel 9.3], deler input op i grupper af 5 elementer, finder rekursivt medianen af gruppernes medianer, bruger det fundne element som pivot til en opdeling, og kalder rekursivt på en af de to dele. Udførselstiden kan beskrives ved følgende rekursionsligning (afrunding ignoreres):

$$T(n) = T(n/5) + T(7n/10) + c \cdot n$$

Hvad bliver udførselstiden hvis man ændrer grupperne i algoritmen til at have størrelse 3 istedet for 5?

The selection algorithm to find the i th smallest element in an unsorted list in worst-case time $O(n)$ [CLRS, Chapter 9.3], divides the input into groups of 5 elements, recursively finds the median of the groups' medians, uses the found element as a pivot for a partition, and recursively calls on one of the two parts. The execution time can be described by the following recurrence relation (ignoring rounding):

$$T(n) = T(n/5) + T(7n/10) + c \cdot n$$

What will the execution time be if you change the groups in the algorithm to have size 3 instead of 5?

$$\Theta(\log n) \quad \Theta(n) \quad \Theta(n \log n) \quad \Theta(n^2)$$

☐ A ☐ B ☒ C ☐ D

Opgave/Problem 292 (4 %)

For en binær max-heap af størrelse n , angiv best-case og worst-case udførselstid for følgende operationer.

For a binary max-heap of size n , give the best-case and worst-case execution time for the following operations.

	$\Theta(1)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	
INSERT, worst-case	☐ A	☒ B	☐ C	☐ D	292.1
INSERT, best-case	☒ A	☐ B	☐ C	☐ D	292.2
HEAP-EXTRACT-MAX, worst-case	☐ A	☒ B	☐ C	☐ D	292.3
HEAP-EXTRACT-MAX, best-case	☒ A	☐ B	☐ C	☐ D	292.4
BUILD-MAX-HEAP, worst-case	☐ A	☐ B	☒ C	☐ D	292.5
BUILD-MAX-HEAP, best-case	☐ A	☐ B	☒ C	☐ D	292.6

Opgave/Problem 293 (4 %)

For QUICKSORT på input af størrelse n , og et givet element e i inputet, hvor mange sammenligninger vil dette element e indgå i under udførslen af QuickSort? Forventet antal sammenligninger er her forventet antal sammenligninger for en tilfældig permutation af input.

For QUICKSORT on input of size n , and a given element e in the input, how many comparisons will this element e be involved in during the execution of QuickSort? The expected number of comparisons here is the expected number of comparisons for a random permutation of the input.

	$\Theta(1)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	
Worst-case antal sammenligninger <i>Worst-case number of comparisons</i>	☐ A	☐ B	☒ C	☐ D	☐ E	293.1
Best-case antal sammenligninger <i>Best-case number of comparisons</i>	☒ A	☐ B	☐ C	☐ D	☐ E	293.2
Forventet antal sammenligninger <i>Expected number of comparisons</i>	☐ A	☒ B	☐ C	☐ D	☐ E	293.3

Opgave/Problem 294 (4 %)

Hvilke af følgende udsagn er sande for alle binære søgetræer med n elementer.

Which of the following statements are true for all binary search trees with n elements.

	Ja/Yes	Nej/No	
Elementet i et blad er altid $\leq$ elementet i roden <i>The element in a leaf is always $\leq$ the element in the root</i>	☐ A	☒ B	294.1
Elementer med samme dybde i træet er sorteret fra venstre-mod-højre <i>Elements at the same depth in the tree are sorted from left to right</i>	☒ A	☐ B	294.2
Længste rod-til-blad sti indeholder $O(\log n)$ knuder <i>Longest root-to-leaf path contains $O(\log n)$ nodes</i>	☐ A	☒ B	294.3
Det højre barn til en knude med rang r har rang $r + 1$ <i>The right child of a node with rank r has rank $r + 1$</i>	☐ A	☒ B	294.4
Langs stien fra roden til det største element, er elementerne voksende <i>Along the path from the root to the largest element, the elements are increasing</i>	☒ A	☐ B	294.5

Opgave/Problem 295 (4 %)

Betragt en $n \times n$ matrix M af heltal, hvor alle rækker og søjler er voksende, dvs. $M[i, j] \leq M[i', j']$ for alle $1 \leq i \leq i' \leq n$ og $1 \leq j \leq j' \leq n$.

Hvad er den bedste worst-case tid man kan opnå for at søge efter et heltal i M ?

Consider an $n \times n$ matrix M of integers, where all rows and columns are sorted, i.e., $M[i, j] \leq M[i', j']$ for all $1 \leq i \leq i' \leq n$ and $1 \leq j \leq j' \leq n$.

What is the best worst-case time one can achieve to search for an integer in M ?

$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$	$O(n\sqrt{n})$	$O(\sqrt{n})$	$O((\log n)^2)$
☐ A	☒ B	☐ C	☐ D	☐ E	☐ F	☐ G

Opgave/Problem 296 (4 %)

Betragt en kø implementeret i et cyklisk array Q , hvor $Q.head$ angiver hovedet af køen og $Q.tail$ er den næste ledige plads i køen (som beskrevet i [CLRS]).

Hvilke af følgende udtryk beregner korrekt størrelsen af køen?

Consider a queue implemented in a circular array Q , where $Q.head$ indicates the head of the queue and $Q.tail$ is the next free space in the queue (as described in [CLRS]).

Which of the following expressions correctly computes the size of the queue?

	Ja/Yes	Nej/No	
$Q.tail - Q.head$	☐ A	☒ X	296.1
$ Q.tail - Q.head $	☐ A	☒ X	296.2
$ Q.tail - Q.head \bmod Q.length$	☐ A	☒ X	296.3
$(Q.tail - Q.head + Q.length - 1) \bmod Q.length$	☐ A	☒ X	296.4
$(Q.tail - Q.head + Q.length) \bmod Q.length$	☒ X	☐ B	296.5

Opgave/Problem 297 (4 %)

Betragt et vilkårligt binært søgetræ, og lad k være et element i et blad i søgetræet. Lad B være alle elementerne på stien fra roden ned til k , og lad A være alle elementer til venstre for stien, og C alle elementer til højre for stien. Gælder følgende udsagn altid for alle $a \in A$, $b \in B$ og $c \in C$?

Consider an arbitrary binary search tree, and let k be an element in a leaf of the search tree. Let B be all elements on the path from the root down to k , and let A be all elements to the left of the path, and C all elements to the right of the path. Does the following statement always hold for all $a \in A$, $b \in B$ and $c \in C$?

	Ja/Yes	Nej/No	
$a \leq b$	☐ A	☒ X	297.1
$b \leq c$	☐ A	☒ X	297.2
$a \leq c$	☒ X	☐ B	297.3

Opgave/Problem 298 (4 %)

Givet to søgetræer T_1 og T_2 indeholdende de samme n elementer, så kan T_1 transformeres til T_2 ved en række rotationer. Hvad er worst-case antallet af rotationer der er krævet for at transformere et søgetræ med n elementer til et andet søgetræ T_2 indeholdende de samme n elementer?

Given two search trees T_1 and T_2 containing the same n elements, T_1 can be transformed into T_2 by a series of rotations. What is the worst-case number of rotations required to transform a search tree with n elements into another search tree T_2 containing the same n elements?

$\Theta(\log n)$ $\Theta(n)$ $\Theta(n \log n)$ $\Theta(n^2)$

☐ A ☒ X ☐ C ☐ D

Opgave/Problem 299 (4 %)

Angiv for hver af nedenstående summer deres værdi i Θ -notationen. Det antages at n er en potens af to.

For each of the sums below, give their value in Θ -notation. It is assumed that n is a power of two.

	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	
$\sum_{i=1}^n i = 1 + 2 + 3 + \dots + n$	☐ A	☐ B	☒ C	299.1
$\sum_{i=0}^{\log n} 2^i = 1 + 2 + 4 + 8 + \dots + n$	☒ A	☐ B	☐ C	299.2
$\sum_{i=1}^{\log n} i \frac{n}{2^i} = 1 \frac{n}{2^1} + 2 \frac{n}{2^2} + 3 \frac{n}{2^3} + \dots + \log n \frac{n}{2^{\log n}}$	☒ A	☐ B	☐ C	299.3
$\sum_{i=1}^n \log i = \log 1 + \log 2 + \log 3 + \dots + \log n$	☐ A	☒ B	☐ C	299.4
$\sum_{i=0}^{\log n} \frac{n}{2^i} = n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \dots + \frac{n}{n}$	☒ A	☐ B	☐ C	299.5

Opgave/Problem 300 (4 %)

Betragt varianten af den binære max-heap, hvor hver knude har d børn istedet for to, og hvor $d \geq 2$ er en parameter. Hvad er tiden for INSERT og EXTRACT-MAX udtrykt som funktion af n og d .

Consider the variant of the binary max-heap where each node has d children instead of two, where $d \geq 2$ is a parameter. What is the time for INSERT and EXTRACT-MAX expressed as a function of n and d .

	$\Theta(d)$	$\Theta(\log_2 n)$	$\Theta(\log_d n)$	$\Theta(d \log_2 n)$	$\Theta(d \log_d n)$	
INSERT	☐ A	☐ B	☒ C	☐ D	☐ E	300.1
EXTRACT-MAX	☐ A	☐ B	☐ C	☐ D	☒ E	300.2

Opgave/Problem 301 (4 %)

Betragt en binær max-heap med $n = 2^k - 1$ forskellige elementer, for et heltal $k \geq 1$.

Hvor mange forskellige knuder i max-heapen kan det mindste element være placeret i?

Consider a binary max-heap with $n = 2^k - 1$ different elements, for an integer $k \geq 1$.

How many different nodes in the max-heap can the smallest element be located in?

1 n k $k - 1$ 2^{k-1} $2^k - 1$

☐ A ☐ B ☐ C ☐ D ☒ E ☐ F

Opgave/Problem 302 (4 %)

Hvor mange gange kaldes RANDOMIZED-PARTITION i RANDOMIZED-SELECT på et array af størrelse n ?

How many times is RANDOMIZED-PARTITION called in RANDOMIZED-SELECT on an array of size n ?

	$\Theta(1)$	$\Theta(\log n)$	$\Theta(\sqrt{n})$	$\Theta(n)$	$\Theta(n \log n)$	
Forventede antal kald <i>Expected number of calls</i>	☐ A	☒ B	☐ C	☐ D	☐ E	302.1
Færrest antal kald (best case) <i>Fewest number of calls (best case)</i>	☒ A	☐ B	☐ C	☐ D	☐ E	302.2
Flest antal kald (worst case) <i>Most number of calls (worst case)</i>	☐ A	☐ B	☐ C	☒ D	☐ E	302.3

Opgave/Problem 303 (4 %)

Lad v være en knude i et rød-sort søgetræ, og antag der er n elementer i v 's venstre undertræ. Hvor mange elementer kan der så maksimalt være i v 's højre undertræ, dvs. hvor ubalanceret kan en knude v være i et rød-sort søgetræ?

Let v be a node in a red-black search tree, and assume there are n elements in v 's left subtree. How many elements can there be at most in v 's right subtree, i.e., how unbalanced can a node v be in a red-black search tree?

$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n\sqrt{n})$	$\Theta(n^2)$	$\Theta(2^n)$
☐ A	☐ B	☐ C	☒ D	☐ E

Opgave/Problem 304 (4 %)

Antag et array $A[1..n]$ repræsenterer en binær max-heap indeholdende n elementer. Hvor hurtigt kan man konstruere et søgetræ (ikke nødvendigvis balanceret) indeholdende elementerne $A[1..n]$?

Assume an array $A[1..n]$ represents a binary max-heap containing n elements. How quickly can one construct a search tree (not necessarily balanced) containing the elements $A[1..n]$?

$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$
☐ A	☐ B	☒ C	☐ D

Opgave/Problem 305 (4 %)

Antag MERGE-SORT udføres på et input af størrelse n og indeholdende to elementer x og y . Hvad er worst-case antal sammenligninger af x med y under udførelsen af MERGE-SORT?

Assume MERGE-SORT is performed on an input of size n and containing two elements x and y . What is the worst-case number of comparisons of x with y during the execution of MERGE-SORT?

$\Theta(1)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$
☒ A	☐ B	☐ C	☐ D	☐ E

Opgave/Problem 306 (4 %)

Antag vi har en orienteret graf med n knuder og positivt vægtede kanter, hvor vi løbende tilføjer yderligere kanter. Vi ønsker at vedligeholde en afstands-tabel over de korteste afstande mellem alle par af knuder.

Hvad er den bedste worst-case tid man kan opnå for at opdatere afstands-tabellen, når man tilføjer en ny kant med positiv vægt til grafen?

Assume we have a directed graph with n nodes and positively weighted edges, where we continuously add additional edges. We want to maintain a distance table of the shortest distances between all pairs of nodes.

What is the best worst-case time we can achieve to update the distance table, when adding a new edge with positive weight to the graph?

$\Theta(1)$ $\Theta(\sqrt{n})$ $\Theta(n)$ $\Theta(n \cdot \log n)$ $\Theta(n \cdot \sqrt{n})$ $\Theta(n^2)$ $\Theta(n^3)$

☐ A

☐ B

☐ C

☐ D

☐ E

☒ F

☐ G