Opgave 1 (4 %)

	Ja	Nej
n er $O(n\sqrt{n})$?	☒	☐
$3n^2$ er $O(n+n^3)$?	☒	☐
$8(2^n)$ er $O(4^n)$?	☒	☐
$7\log n$ er $O(n^{1/7})$?	☒	☐
$7n^7+7n$ er $O(14n)$?	☐	☒

Opgave 2 (4 %)

Opskriv følgende funktioner efter stigende orden med hensyn til O-notationen:

$$(\log n)^2$$
$$2^n$$
$$1/n$$
$$(\log n)^n$$
$$n^2$$

Svar: $1/n$ $\quad (\log n)^2$ $\quad n^2$ $\quad 2^n$ $\quad (\log n)^n$

Opgave 3 (4 %)

Angiv for hver af nedenstående summer deres værdi i Θ-notationen. Det antages at n er en potens af to.

	$\Theta(n)$	$\Theta(n\log n)$	$\Theta(n^2)$
$\sum_{i=1}^{n} i = 1+2+3+\cdots+n$	☐	☐	☒
$\sum_{i=0}^{\log n} 2^i = 1+2+4+8+\cdots+n$	☒	☐	☐
$\sum_{i=1}^{\log n} i\frac{n}{2^i} = 1\frac{n}{2^1}+2\frac{n}{2^2}+3\frac{n}{2^3}\cdots+\log n\frac{n}{2^{\log n}}$	☒	☐	☐
$\sum_{i=1}^{n} \log i = \log 1+\log 2+\log 3+\cdots+\log n$	☐	☒	☐
$\sum_{i=0}^{\log n} \frac{n}{2^i} = n+\frac{n}{2}+\frac{n}{4}+\frac{n}{8}\cdots+\frac{n}{n}$	☒	☐	☐

Opgave 4 (4 %)

Angiv for hver af nedenstående algoritmer udførselstiden som funktion af n i O-notation.

Algoritme Loop1(n)
  for $i = 1$ **to** n
    $j = i$
    while $j > 0$
      $j = j - 1$

Algoritme Loop2(n)
  $i = 1$
  while $i < n * n$
    $i = 2 * i$

Algoritme Loop3(n)
  for $i = 1$ **to** n
    for $j = 1$ **to** i
      $k = 1$
      while $k \leq i + j$
        $k = 2 * k$

Svar Loop1: $O(n^2)$

Svar Loop2: $O(\log n)$

Svar Loop3: $O(n^2 \cdot \log n)$

Opgave 5 (4 %)

Angiv for hver af nedenstående algoritmer udførselstiden som funktion af n i O-notation.

Algoritme Loop1(n)
  $i = 1$
  $j = 1$
  while $i < n$
    while $j < i$
      $j = j + 1$
    $i = i * 2$

Algoritme Loop2(n)
  $i = 1$
  while $i < n$
    $j = 1$
    while $j < i$
      $j = j + 1$
    $i = i * 2$

Algoritme Loop3(n)
  $i = 2$
  while $i < n$
    $i = i * i$

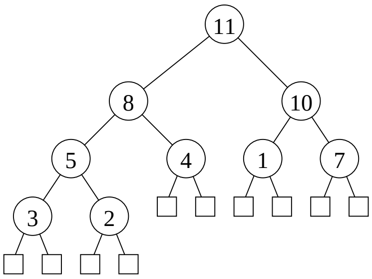
Svar Loop1: $O(n)$

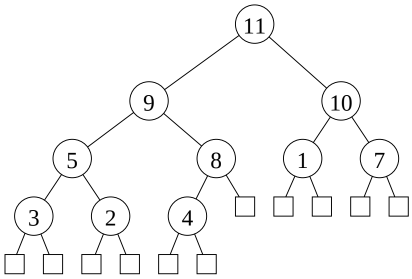
Svar Loop2: $O(n)$

Svar Loop3: $O(\log \log n)$

Opgave 6 (4 %)

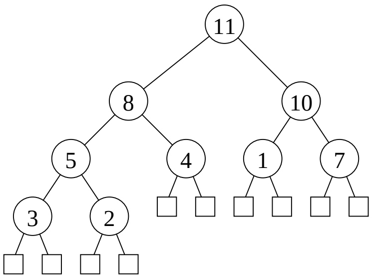
Tegn hvordan nedenstående binære max-heap ser ud efter indsættelse af elementet 9.

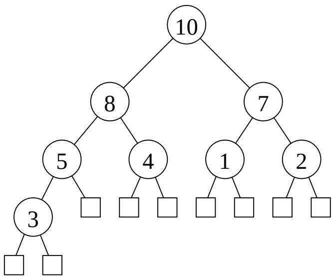




Svar: ______________________

Tegn hvordan nedenstående binære max-heap ser ud efter en Heap-Extract-Max operation.

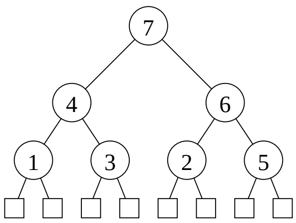




Svar: ______________________

Opgave 7 (4 %)

Tegn den binære max-heap efter indsættelse af elementerne 1, 3, 2, 5, 4, 6, og 7 i den givne rækkefølge, startende med den tomme heap.



Svar: ______________________

Opgave 8 (4 %)

Angiv hvordan nedenstående array ser ud efter anvendelsen af Build-Max-Heap for arrayet.

1	2	3	4	5	6	7	8	9	10
1	5	9	3	7	2	6	10	4	8

1	2	3	4	5	6	7	8	9	10
10	8	9	5	7	2	6	3	4	1

Svar: ______________________

(Opgavesættet fortsætter)

Opgave 9 (4 %)

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

2300 4123 3323 4411 1123 0000

Svar: 2300 0000 4411 4123 3323 1123

Opgave 10 (4 %)

Angiv resultatet af at anvende PARTITION(A,5,13) på nedenstående array.

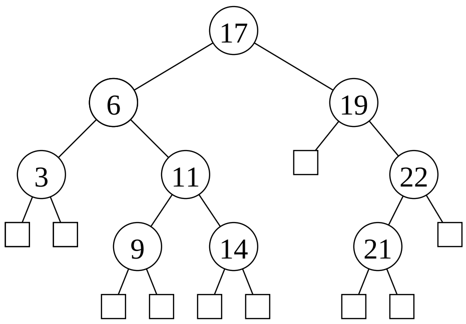
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	8	16	1	6	2	4	13	17	15	3	5	18	9	11	24	12

Svar:

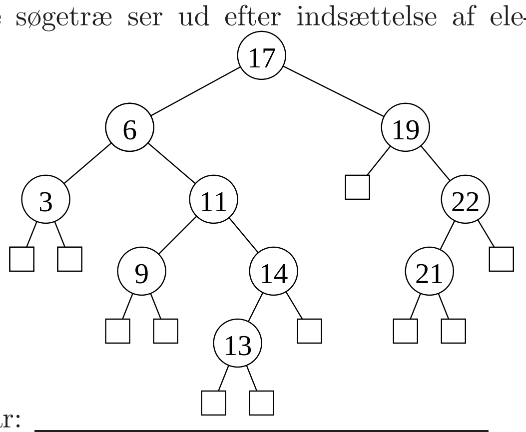
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
8	16	1	6	**2**	**4**	**3**	**5**	**9**	**13**	**17**	**18**	**15**	11	24	12

Opgave 11 (4 %)

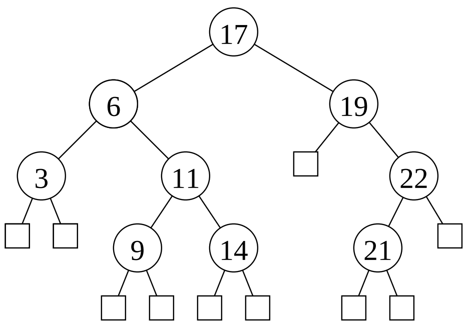
Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter indsættelse af elementet 13.



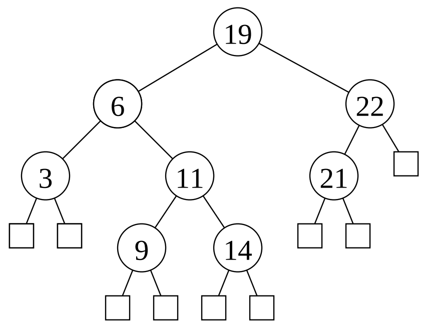
Svar:



Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter slettelse af elementet 17.

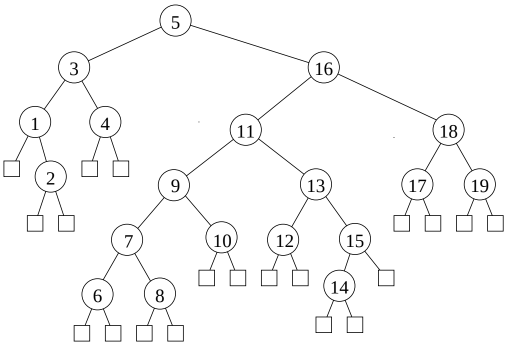


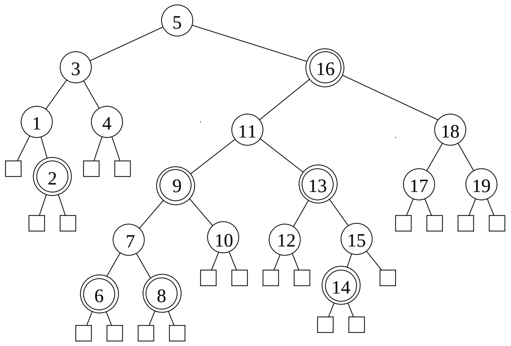
Svar:



Opgave 12 (4 %)

Angiv hvorledes knuderne i nedenstående binære søgetræ kan farves røde og sorte, således at det resulterende træ er et lovligt rød-sort træ.

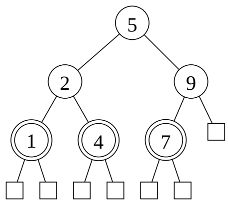


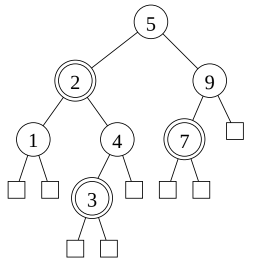


Svar: ______________________________

Opgave 13 (4 %)

Tegn hvordan nedenstående rød-sorte træ (dobbeltcirkler angiver røde knuder) ser ud efter indsættelse af elementet 3.

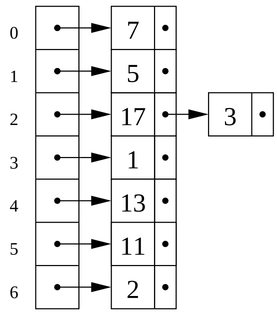




Svar: ______________________________

Opgave 14 (4 %)

Tegn en hashtabel hvor der anvendes kædede lister til at håndtere kollisioner, når hashfunktionen er $h(k) = 3k \bmod 7$ og der indsættes elementerne 1, 2, 3, 5, 7, 11, 13, og 17 i den givne rækkefølge.



Svar: ______________________________

Opgave 15 (4 %)

Tegn hvordan en hashtabel der anvender *linear probing* ser ud efter at elementerne 3, 0, 8, 2, 4, 12 og 11 indsættes i den givne rækkefølge, når hashfunktionen er $h(k) = 2k \bmod 9$.

0	1	2	3	4	5	6	7	8

Svar:

0	1	2	3	4	5	6	7	8
0	12			2	11	3	8	4

Opgave 16 (4 %)

Tegn hvordan en hashtabel der anvender *kvadratisk hashing* ser ud efter at elementerne 2, 3, 7, 5 og 11 indsættes i den givne rækkefølge, når hashfunktionen er

$$h(k,i) = \left(h'(k) + i + 3i^2\right) \bmod 7$$

$$h'(k) = 2k \bmod 9$$

0	1	2	3	4	5	6

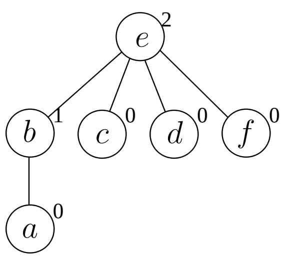
Svar:

0	1	2	3	4	5	6
11	5			2	7	3

Opgave 17 (4 %)

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

makeset(a)
makeset(b)
makeset(c)
makeset(d)
makeset(e)
makeset(f)
union(a,b)
union(a,c)
union(d,e)
union(c,e)
union(c,f)

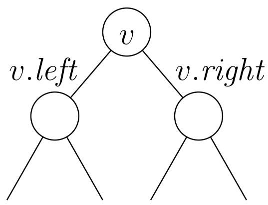


Svar:

Opgave 18 (4 %)

En sorteret liste af reelle tal $x_1 < x_2 < \cdots < x_n$ er *dense* hvis $x_{i+1} - x_i \leq 1$ for alle $i = 1..n-1$. F.eks. er listen 1.0, 1.5, 1.8, 2.9, 3.7 ikke dense, da $2.9 - 1.8 > 1$.

Betragt et søgetræ hvor hver knude v gemmer en værdi $v.x$, og knuderne er ordnet venstre-mod-højre efter stigende værdi. I hver knude gemmes ud over $v.x$ også $v.min$ og $v.max$, som er hhv. den mindste og største værdi i v's undertræ, og en boolsk variabel $v.dense$ der angiver om listen af punkter i v's undertræ er dense. Angiv hvorledes disse værdier kan beregnes når den tilsvarende information er kendt ved de to børn $v.left$ og $v.right$ (det kan antages at disse begge eksisterer).



Svar $v.min$ = $v.left.min$

Svar $v.max$ = $v.right.max$

Svar $v.dense$ = $v.left.dense \wedge v.right.dense \wedge v.x - v.left.max \leq 1.0 \wedge v.right.min - v.x \leq 1.0$

Opgave 19 (4 %)

Angiv de tre egenskaber en termineringsfunktion μ skal have:

Svar : heltallig

Svar : ikke-negativ

Svar : strengt aftagende

Transitionssystem Nedtælling
Konfigurationer: $\{[m, s] \mid \text{heltal } m, s \ \wedge \ m \geq 0 \ \wedge \ 0 \leq s \leq 59\}$

$[m, s] \triangleright [m, s-1] \quad \textbf{if} \quad s > 0$
$[m, s] \triangleright [m-1, 59] \quad \textbf{if} \quad m > 0 \ \wedge \ s = 0$

Opgave 20 (4 %)

For hvert af nedenstående udsagn, angiv om de er en invariant for ovenstående transitionssystem Nedtælling. Startkonfigurationen antages at være $[n, 0]$ hvor $n \geq 0$.

	Ja	Nej
$m + s \leq n$	☐	☒
$m \leq n$	☒	☐
$59m + s \leq 59n$	☒	☐
$m \geq s$	☐	☒
$60m + s \leq 60n$	☒	☐

Opgave 21 (4 %)

For hver af nedenstående funktioner, angiv om de er en termineringsfunktion for ovenstående transitionssystem Nedtælling.

	Ja	Nej
$\mu(m, s) = m + s$	☐	☒
$\mu(m, s) = 59m + s$	☐	☒
$\mu(m, s) = 60m + s$	☒	☐
$\mu(m, s) = m + s/60$	☐	☒
$\mu(m, s) = 59 \cdot 60 \cdot m + s^2$	☒	☐

Lad $\#_1(x)$ betegne antal bits lig 1 i den binære repræsentation af et ikke negativt heltal x. F.eks. er $\#_1(14) = 3$ da $14 = 1110_2$.

Algoritme BitsSet(n)
Inputbetingelse : Heltal $n \geq 0$
Outputkrav : $r = \#_1(n)$
Metode : $x \leftarrow n$;
  $r \leftarrow 0$;
  $\{I\}$ **while** $x > 0$ **do**
    if x ulige **then**
      $x \leftarrow x - 1$;
      $r \leftarrow r + 1$
    else
      $x \leftarrow x/2$

Opgave 22 (4 %)

For hvert af nedenstående udsagn, angiv om de er en invariant I for ovenstående algoritme BitsSet.

	Ja	Nej
$0 < x \leq n$	☐	☒
$\#_1(x) = \#_1(n)$	☐	☒
$\#_1(x) + r = \#_1(n)$	☒	☐
$r = \#_1(x)$	☐	☒
$r \leq x$	☐	☒

Opgave 23 (4 %)

For hver af nedenstående funktioner, angiv om de er en termineringsfunktion for ovenstående algoritme BitsSet.

	Ja	Nej
$\mu(n, x, r) = n$	☐	☒
$\mu(n, x, r) = x$	☒	☐
$\mu(n, x, r) = \#_1(x)$	☐	☒
$\mu(n, x, r) = \#_1(x) - r$	☐	☒
$\mu(n, x, r) = \#_1(n) - \#_1(x) - r$	☐	☒

(Opgavesættet fortsætter)

Opgave 24 (4 %)

Givet et positivt heltal n, så beregner nedenstående algoritme n^3. For at vise gyldigheden af algoritmen skal I_i, I_s og I_r være invarianter omkring variablerne i, s og r. Angiv invarianter hvormed gyldigheden af algoritmen kan bevises (bevis for invarianterne kræves ikke).

Algoritme Power3(n)
Inputbetingelse : Heltal $n \geq 1$
Outputkrav : $r = n^3$
Metode : $i \leftarrow 1$;
$s \leftarrow 1$;
$r \leftarrow 1$;
$\{I_i \wedge I_s \wedge I_r\}$ **while** $i < n$ **do**
$i \leftarrow i + 1$;
$s \leftarrow s + 2i - 1$;
$r \leftarrow r + 3s - 3i + 1$

Svar I_i: $1 \leq i \leq n$

Svar I_s: $s = i^2$

Svar I_r: $r = i^3$

For at kunne bevise at algoritmen terminerer, kræves en passende termineringsfunktion. Angiv en termineringsfunktion (bevis for at termineringsfunktionen har de nødvendige egenskaber kræves ikke).

Svar μ: $n - i$

Opgave 25 (4 %)

Rød-sorte søgetræer understøtter INSERT og DELETE på et træ med n elementer i worst-case tid $O(\log n)$. Med en passende potentialefunktion kan man argumentere for at INSERT tager amortiseret tid $O(\log n)$ og DELETE tager amortiseret $O(1)$ tid. Angiv en sådan potentialefunktion.

Svar Φ = $n \log n$