Opgave 1 (4 %)

	Ja	Nej
n er $O(n^{1/2})$?	☐	☒
$3n^7 + n^5$ er $O(n^6)$?	☐	☒
$(\log n)^2$ er $O(2^{\log n})$?	☒	☐
n^1 er $O(1)$?	☐	☒
$1/n$ er $O(\log n)$?	☒	☐

Opgave 2 (4 %)

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

$$2^{3\log n}$$
$$n^{3/2}$$
$$\sqrt{n}$$
$$n \cdot \log n$$
$$(3/2)^n$$

Svar: $\sqrt{n}$ $\quad n \cdot \log n$ $\quad n^{3/2}$ $\quad 2^{3\log n}$ $\quad (3/2)^n$

Opgave 3 (4 %)

Antag vi ønsker at understøtte operationen PRED(x) på nedenstående fire datastrukturer, som returnerer det største element i datastrukturen mindre end x. F.eks. for mængden $\{1, 2, 4, 7, 8\}$ er PRED$(5) = 4$. Angiv for hver af datastrukturerne hvor lang tid det vil tage at udføre PRED(x).

	$O(1)$	$O(\log n)$	$O(n)$
Enkelt-kædet sorteret liste ?	☐	☐	☒
Dobbelt-kædet sorteret liste ?	☒	☐	☐
Binær min-heap ?	☐	☐	☒
Rød-sort søge-træ ?	☐	☒	☐

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 $j \geq 1$
      $j = j - 1$

Algoritme Loop2(n)
  $s = 0$
  while $n > 1$
    while $n > 1$
      $n = \lfloor n/2 \rfloor$
    $n = \lfloor n/2 \rfloor$

Algoritme Loop3(n)
  $s = 0$
  for $i = 1$ **to** n
    for $j = 1$ **to** n
      for $k = 1$ **to** n
        $s = s + 1$

Svar Loop1: $O(n^2)$

Svar Loop2: $O(\log n)$

Svar Loop3: $O(n^3)$

Opgave 5 (4 %)

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

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

Algoritme Loop2(n)
  while $n > 0$
    $n = \lfloor n/2 \rfloor$

Algoritme Loop3(n)
  $i = 1$
  $p = 1$
  while $p < n$
    $i = i + 1$
    $p = p * i$

Svar Loop1: $O(n)$

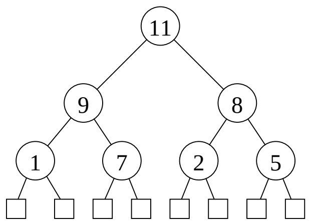
Svar Loop2: $O(\log n)$

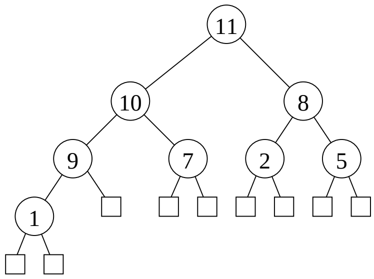
Svar Loop3: $O(\frac{\log n}{\log \log n})$

(Opgavesættet fortsætter)

Opgave 6 (4 %)

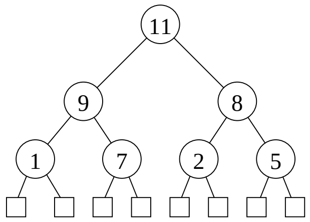
Tegn hvordan nedenstående binære max-heap ser ud efter indsættelse af elementet 10.

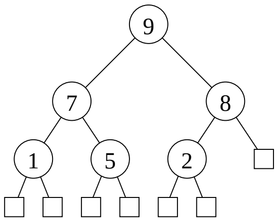




Svar:

Tegn hvordan nedenstående binære max-heap ser ud efter en HEAP-EXTRACT-MAX operation.

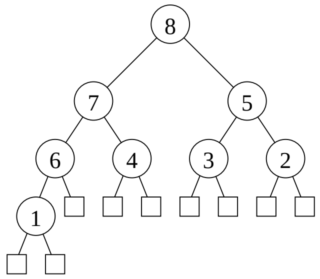




Svar:

Opgave 7 (4 %)

Tegn den binære max-heap efter indsættelse af elementerne 1, 3, 6, 4, 8, 5, 2, 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
5	2	4	1	10	7	3	6	8	9

Svar:

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

Opgave 9 (4 %)

Betragt Radix-Sort anvendt på nedenstående liste af tal ($d = 4$, $k = 7$). Angiv den delvist sorterede liste efter at radix-sort har sorteret tallene efter de *to* mindst betydende cifre.

1431 4123 6310 3131 5223 2531

Svar: 6310 4123 5223 1431 3131 2531

Opgave 10 (4 %)

Angiv resultatet af at anvende Partition(A,5,10) på nedenstående array.

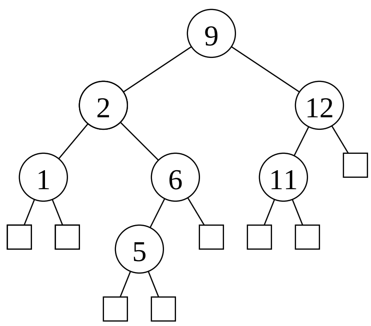
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	2	14	10	6	12	5	11	13	2	8	17	1	3	12	7	9

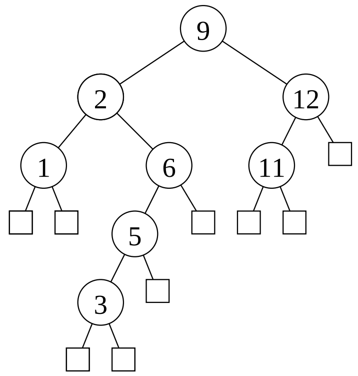
Svar:

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

Opgave 11 (4 %)

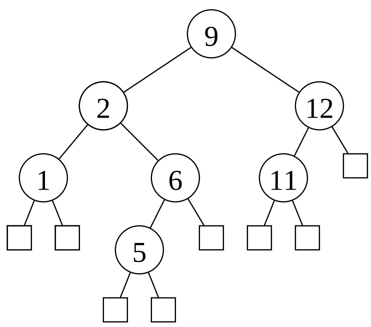
Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter indsættelse af elementet 3.

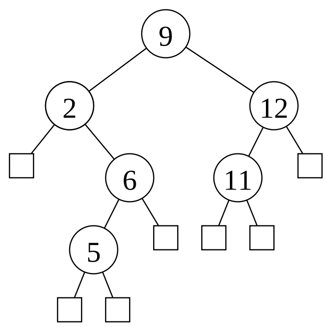




Svar:

Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter slettelse af elementet 1.

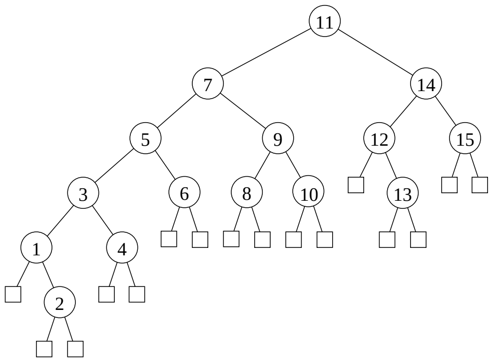


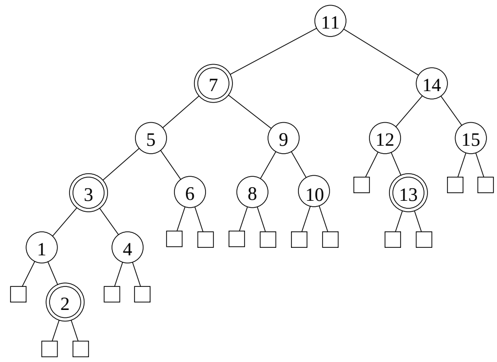


Svar:

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.

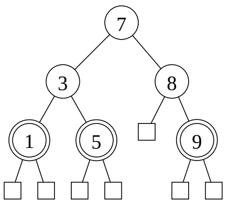


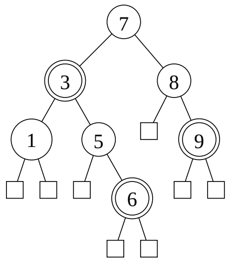


Svar: ______________________________

Opgave 13 (4 %)

Tegn hvordan nedenstående rød-sorte træ (dobbeltcirkler angiver røde knuder) ser ud efter indsættelse af elementet 6.

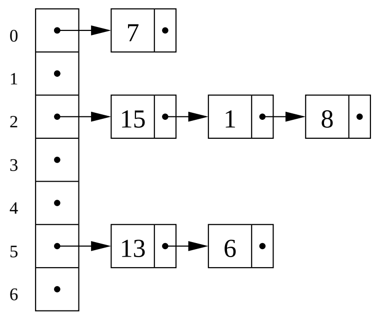




Svar: ______________________________

Opgave 14 (4 %)

Tegn en hashtabel hvor der anvendes kædede lister til at håndtere kollisioner, når hashfunktionen er $h(k) = 2k \bmod 7$ og der indsættes elementerne 8, 6, 1, 15, 7 og 13 i den givne rækkefølge.



Svar: ______________________________

Opgave 15 (4 %)

Tegn hvordan en hashtabel der anvender *linear probing* ser ud efter at elementerne 8, 6, 1, 15, 13 og 2 indsættes i den givne rækkefølge, når hashfunktionen er $h(k) = 2k \bmod 7$.

0	1	2	3	4	5	6

0	1	2	3	4	5	6
2		8	1	15	6	13

Svar: ______________________

Opgave 16 (4 %)

Tegn hvordan en hashtabel der anvender *quadratic hashing* ser ud efter at elementerne 1, 6, 8, 2, 3 og 15 indsættes i den givne rækkefølge, når hashfunktionen er

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

0	1	2	3	4	5	6

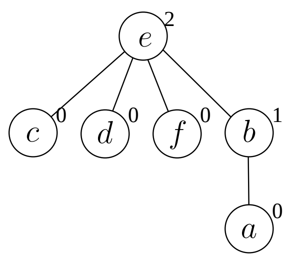
0	1	2	3	4	5	6
15	3	1		8	6	2

Svar: ______________________

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(a,d)
union(c,f)



Svar: ______________________

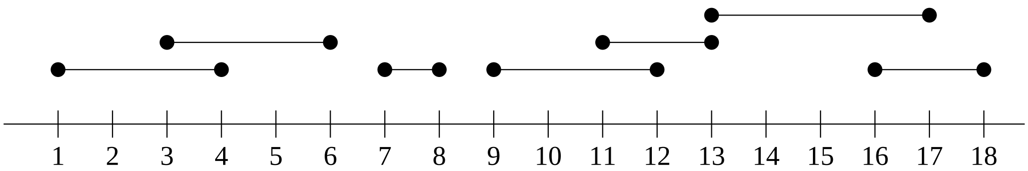
Opgave 18 (4 %)

Betragt radix-sort anvendt til sortering af n tal, hvert tal bestående af w bits, og hvor hver iteration af counting-sort sorterer efter b bits. Hvad er udførselstiden af radix-sort ?

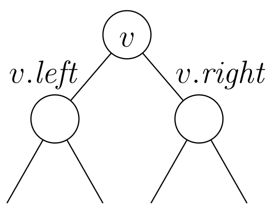
Svar: $O(w/b \cdot (n + 2^b))$

Opgave 19 (4 %)

For en mængde af intervaller, betegner vi med R det længste interval, der er foreningen af en delmængde af intervallerne og som indeholder det højreste endepunkt af alle intervallerne. F.eks. for nedenstående mængde af intervaller er $R = [9, 18]$.



Betragt et søgetræ hvor hver knude v gemmer et interval $[v.\ell, v.r]$ med endepunkter $v.\ell$ and $v.r$, og hvor intervallerne er sorteret fra venstre-mod-højre i søgetræet efter deres venstre endepunkt. I hver knude gemmes ud over $v.\ell$ og $v.r$ også $v.R\ell$ og $v.Rr$, således at $[v.R\ell, v.Rr]$ er R-intervallet for intervallerne gemt i v's undertræ. Angiv hvorledes disse værdier kan beregnes når den tilsvarende information er kendt ved de to børn $v.\mathit{left}$ og $v.\mathit{right}$ (det kan antages at disse begge eksisterer).



Svar $v.Rr$ $=$ $\max\{v.r, v.\mathit{left}.Rr, v.\mathit{right}.Rr\}$

Svar $v.R\ell$ $=$

$$\begin{cases} v.\mathit{left}.R\ell & \text{hvis } v.\mathit{left}.Rr \geq v.\ell \land \max\{v.r, v.\mathit{left}.Rr\} \geq v.\mathit{right}.R\ell \\ v.\ell & \text{hvis } v.\mathit{left}.Rr < v.\ell \land v.r \geq v.\mathit{right}.R\ell \\ v.\mathit{right}.R\ell & \text{hvis } \max\{v.\mathit{left}.Rr, v.r\} < v.\mathit{right}.R\ell \end{cases}$$

Transitionssystem Frem-og-Tilbage
Konfigurationer: $\{2^i \cdot 3^k \mid \text{ hvor } i, k \text{ er heltal} \wedge i \geq 0 \wedge k \geq 0\}$
$2^i \cdot 3^k \triangleright 2^{i+2} \cdot 3^{k-1}$ **if** $k \geq 1$
$2^i \cdot 3^k \triangleright 2^{i-1} \cdot 3^k$ **if** $i \geq 1$

Opgave 20 (4 %)

For hvert af nedenstående udsagn, angiv om de er en invariant for ovenstående transitionssystem Frem-og-Tilbage. Startkonfigurationen antages at være $n = 2^{i_0} \cdot 3^{k_0}$.

	Ja	Nej
$2^i \cdot 3^k = n$	☐	☒
$2^i \cdot 3^k \leq n$	☐	☒
$2^i \cdot 3^k \leq 4/3 \cdot n$	☐	☒
$i + k \leq \log_2 n$	☐	☒
$i + k \leq 2 \log_3 n$	☒	☐

Opgave 21 (4 %)

For hver af nedenstående funktioner, angiv om de er en termineringsfunktion for ovenstående transitionssystem Frem-og-Tilbage.

	Ja	Nej
$\mu(2^i \cdot 3^k) = 3i + k$	☐	☒
$\mu(2^i \cdot 3^k) = i + k$	☐	☒
$\mu(2^i \cdot 3^k) = i + 3k$	☒	☐
$\mu(2^i \cdot 3^k) = i + 2k$	☐	☒
$\mu(2^i \cdot 3^k) = 2i + 5k$	☒	☐

Algoritme Square(n)
Inputbetingelse : heltal $n \geq 1$
Outputkrav : $r = n^2$
Metode : $i \leftarrow 1;$
$r \leftarrow 1;$
$\{I\}$ **while** $i < n$ **do**
$i \leftarrow i + 1;$
$r \leftarrow r + i$
$r \leftarrow 2 * r - n$

Opgave 22 (4 %)

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

	Ja	Nej
$i < n$	☐	☒
$r = i^2$	☐	☒
$r = n(n+1)/2$	☐	☒
$r = i(i+1)/2$	☒	☐
$r \geq i$	☒	☐

Opgave 23 (4 %)

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

	Ja	Nej
$\mu(n, i, r) = n^2 - i^2$	☒	☐
$\mu(n, i, r) = i$	☐	☒
$\mu(n, i, r) = n - i$	☒	☐
$\mu(n, i, r) = n^2 - r^2$	☐	☒
$\mu(n, i, r) = r - i$	☐	☒

Opgave 24 (4 %)

Givet et positivt heltal n, så beregner nedenstående algoritme $n!$.

Algoritme Factorial(n)
Inputbetingelse : positivt heltal $n \geq 1$
Outputkrav : $r = n! = 1 \cdot 2 \cdot 3 \cdots (n-1) \cdot n$
Metode : $i \leftarrow n$;
$r \leftarrow n$;
$\{I_i \ \wedge \ I_r\}$ **while** $i > 1$ **do**
$i \leftarrow i - 1$;
$r \leftarrow r * i$

For at vise gyldigheden af algoritmen skal I_i og I_r være invarianter omkring variablerne i og r. Angiv invarianter hvormed gyldigheden af algoritmen kan bevises (bevis for invarianterne kræves ikke).

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

Svar I_r: $r = \prod_{j=i}^{n} j$

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: i

Opgave 25 (4 %)

En binær max-heap understøtter Insert og Heap-Extract-Max i worst-case $O(\log n)$ tid. Angiv en potentiale funktion så Insert tager amortiseret $O(\log n)$ tid og Heap-Extract-Max amortiseret $O(1)$ tid.

Svar Φ : $n \cdot \log n$