DATALOGISK INSTITUT, AARHUS UNIVERSITET

Det Naturvidenskabelige Fakultet
EKSAMEN
Grundkurser i Datalogi
Algoritmer og Datastrukturer 1 (2003-ordning)
Antal sider i opgavesættet (incl. forsiden): 12 (tolv)
Eksamensdag: Fredag den 11. august 2006, kl. 9.00-11.00
Eksamenslokale: Trøjborg, Willemoesgade 15, Århus N, 8200 Århus N
Tilladte medbragte hjælpemidler: Alle sædvanlige hjælpemidler (lærebøger og notater)
Materiale der udleveres til eksaminanden:

Årskort ____________________

Navn ____

Skriftlig Eksamen
Algoritmer og Datastrukturer 1 (2003-ordning)

Datalogisk Institut
Aarhus Universitet

Fredag den 11. august 2006, kl. 9.00-11.00

Dette eksamenssæt består af en kombination af små skriftlige opgaver og multiple-choice-opgaver. Opgaverne besvares på opgaveformuleringen **som afleveres**.

For hver opgave er angivet opgavens andel af det samlede eksamenssæt.

For multiple-choice-opgaver gælder følgende. Hvert delspørgsmål har præcist et svar. For hvert delspørgsmål, kan du vælge ét svar ved at afkrydse den tilsvarende rubrik. Et multiple-choice-delspørgsmål bedømmes som følgende:

- Hvis du sætter kryds ved det rigtige svar, får du 1 point.
- Hvis du ikke sætter nogen krydser, får du 0 point.
- Hvis du sætter kryds ved et forkert svar, får du $-\frac{1}{k-1}$ point, hvor k er antal svarmuligheder.

For en multiple-choice-opgave med vægt $v\,\%$ og med n delspørgsmål, hvor du opnår samlet s point, beregnes din besvarelse af multiple-choice-opgaven som:

$$\max\left\{0\,,\,\frac{s}{n}\right\}\,\cdot\,v\,\%$$

(Opgavesættet fortsætter)

Opgave 1 (4 %)

	Ja	Nej
$5n^3$ er $O(n^2)$?	☐	☒
$n \cdot \log n$ er $O(7(\log n)^3)$?	☐	☒
3^n er $O(n^3)$?	☐	☒
$\sqrt{n}$ er $O(n/\log n)$?	☒	☐
$n^2 + n^2$ er $\Omega(n^4)$?	☐	☒

Opgave 2 (4 %)

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

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

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

Opgave 3 (4 %)

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

Algoritme Loop1(n)
  $x \leftarrow 0$
  $i \leftarrow n$
  while $i > 0$ **do**
    for $j \leftarrow 1$ **to** i **do**
      $x \leftarrow x + 1$
    $i \leftarrow i - 1$

Algoritme Loop2(n)
  $x \leftarrow 0$
  $i \leftarrow 1$
  while $i < n$ **do**
    for $j \leftarrow 1$ **to** i **do**
      $x \leftarrow x + 1$
    $i \leftarrow 2 * i$

Algoritme Loop3(n)
  $i \leftarrow 1$
  $j \leftarrow 1$
  while $j < n$ **do**
    $j \leftarrow j + i$
    $i \leftarrow i + 1$

Svar Loop1: $O(n^2)$

Svar Loop2: $O(n)$

Svar Loop3: $O(\sqrt{n})$

Opgave 4 (4 %)

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

Algoritme Loop1(n)
  $x \leftarrow 0$
  for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow 1$ **to** $i * i$ **do**
      $x \leftarrow x + 1$

Algoritme Loop2(n)
  $i \leftarrow n$
  $x \leftarrow 0$
  while $i \geq 1$ **do**
    $j \leftarrow i$
    while $j \geq 1$ **do**
      $x \leftarrow x + 1$
      $j \leftarrow \lfloor j/2 \rfloor$
    $i \leftarrow \lfloor i/2 \rfloor$

Algoritme Loop3(n)
  $i \leftarrow 1$
  while $i \leq n$ **do**
    $k \leftarrow 1$
    $j \leftarrow 1$
    while $j \leq n$ **do**
      $k \leftarrow k + 1$
      $j \leftarrow j + k$
    $i \leftarrow i * 2$

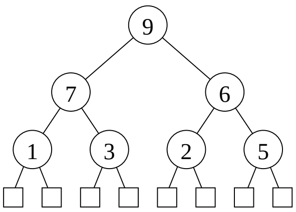
Svar Loop1: $O(n^3)$

Svar Loop2: $O((\log n)^2)$

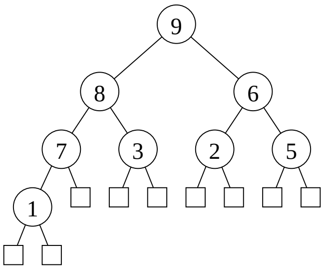
Svar Loop3: $O(\sqrt{n} \log n)$

Opgave 5 (4 %)

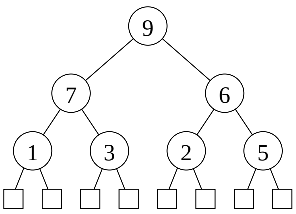
Tegn hvordan nedenstående binære max-heap ser ud efter indsættelse af elementet 8.



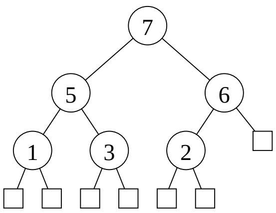
Svar:



Tegn hvordan nedenstående binære max-heap ser ud efter en heap-extract-max operation.

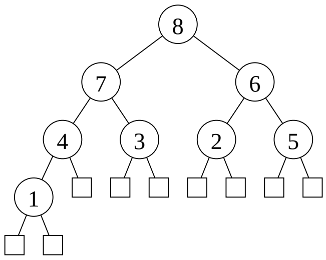


Svar:



Opgave 6 (4 %)

Tegn den binære max-heap efter indsættelse af elementerne 1, 2, 3, 4, 5, 6, 7 og 8 i den givne rækkefølge, startende med den tomme heap.



Svar:

Opgave 7 (4 %)

Angiv hvordan nedenstående array ser ud efter anvendelsen af build-max-heap for arrayet.

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

Svar:

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

Opgave 8 (4 %)

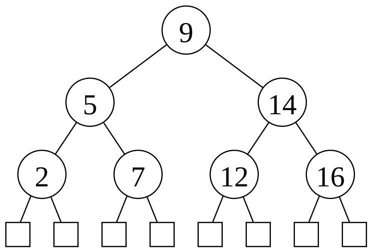
Betragt radix-sort anvendt på nedenstående liste af tal ($d = 4$, $k = 10$). Angiv den delvist sorterede liste efter at radix-sort har sorteret tallene efter de to mindst betydende cifre.

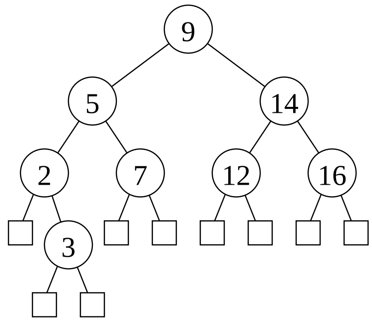
7811 6905 1499 2211 8006 1399

Svar: 6905 8006 7811 2211 1499 1399

Opgave 9 (4 %)

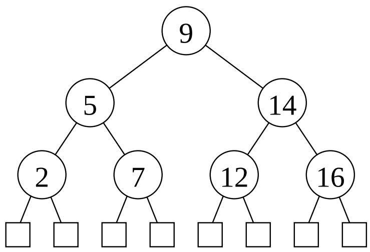
Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter indsættelse af elementet 3.

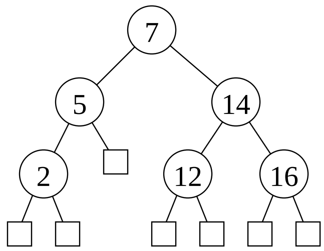




Svar: ____________________________

Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter slettelse af elementet 9.





Svar: ____________________________

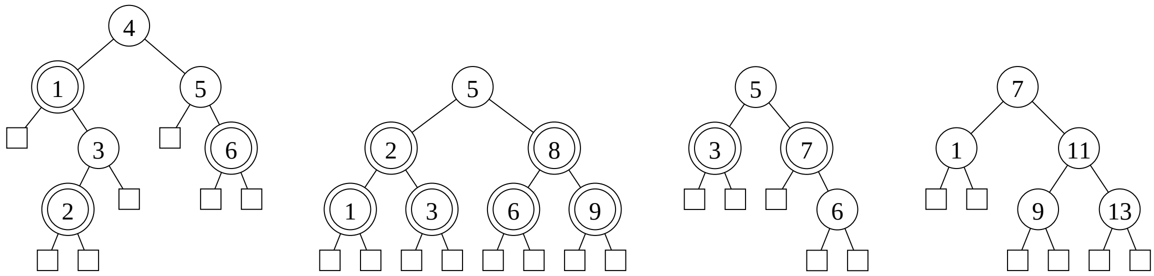
Opgave 10 (4 %)

Afgør for nedenstående udsagn om de er sande for ethvert rød-sort søgetræ med n interne knuder.

	Ja	Nej
Der er højest dobbelt så mange røde knuder som sorte knuder ?	☒	☐
Antal røde knuder på alle rod til blad stier er det samme ?	☐	☒
Der findes en sti fra roden til et blad hvor alle knuder er sorte ?	☐	☒
Antal rotationer under en indsættelse er $O(1)$?	☒	☐
Antal omfarvninger af knuder under en indsættelse er $O(1)$?	☐	☒
Antal blade = antal interne knuder + 1 ?	☒	☐

Opgave 11 (4 %)

Angiv for hvert af nedenstående træer om det er et lovligt søgetræ, et lovligt rød-sort søgetræ, eler ingen af delene (dobbeltcirkler angiver røde knuder).

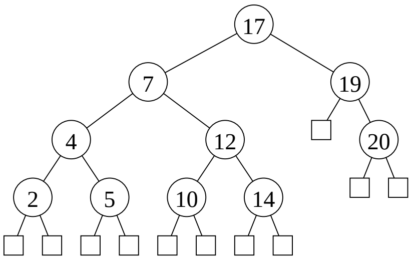


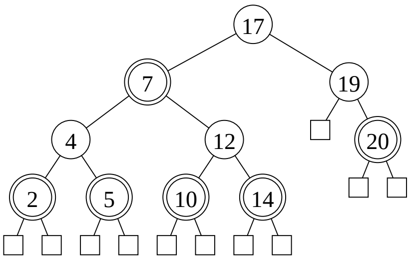
a) b) c) d)

	Rød-sort søgetræ	Søgetræ, men ikke rød-sort	Ikke et søgetræ
a)	☐	☒	☐
b)	☐	☒	☐
c)	☐	☐	☒
d)	☐	☒	☐

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.

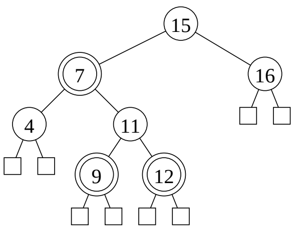


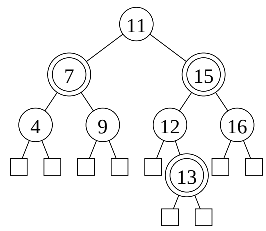


Svar: ______________________

Opgave 13 (4 %)

Tegn hvordan nedenstående rød-sorte træ (dobbeltcirkler angiver røde knuder) ser ud efter indsættelse af elementet 13.



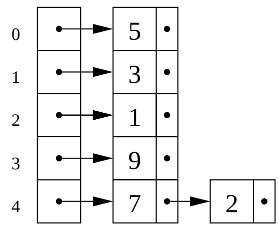


Svar: ______________________

(Opgavesættet fortsætter)

Opgave 14 (4 %)

Tegn en hashtabel hvor der anvendes kædede lister til at håndtere kollisioner, når hashfunktionen er $h(k) = k * 2 \bmod 5$ og der indsættes elementerne 2, 3, 1, 7, 5, og 9 i den givne rækkefølge.



Svar:

Opgave 15 (4 %)

Tegn hvordan en hashtabel, der anvender *linear probing* med hashfunktionen $h(k) = 5 \cdot k \bmod 13$, ser ud efter at elementerne 3, 2, 11, 12, 7, 16, 6, og 1 er indsat i den givne rækkefølge startende med en tom hashtabel.

Svar:

0	1	2	3	4	5	6	7	8	9	10	11	12
		3	11	16	6	1		12	7	2		

Opgave 16 (4 %)

Nedenstående er en hashtabel hvor der er anvendt *double hashing*. Den anvendte hashfunktion er

$$h(k, i) = h_1(k) + i \cdot h_2(k) \bmod 9$$

hvor $h_1(k) = k \bmod 9$ og $h_2(k) = 1 + (k \bmod 7)$.

Tegn hvordan hashtabellen ser ud efter at $k = 9$, $k = 4$ og $k = 14$ indsættes i den givne rækkefølge.

0	1	2	3	4	5	6	7	8
				13				

Svar:

0	1	2	3	4	5	6	7	8
9				13	4	14		

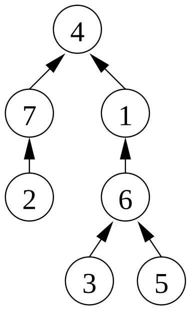
Opgave 17 (4 %)

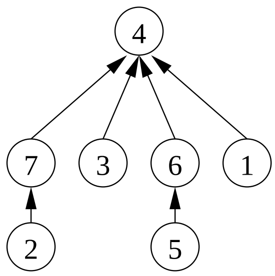
Angiv for hver af nedenstående datastrukturer udførselstiden for indsættelsen af elementerne fra henholdsvis de to sekvenser $1, 2, 3, \ldots, n$ og $n, n-1, n-2, \ldots, 1$, når der altid startes med en tom datastruktur..

Datastruktur	$1, 2, 3, \ldots, n$	$n, n-1, n-2, \ldots, 1$
MaxHeap	$\Theta(n \log n)$	$\Theta(n)$
Sorteret enkelt-kædet liste	$\Theta(n^2)$	$\Theta(n)$
Rød-sort søgetræ	$\Theta(n \log n)$	$\Theta(n \log n)$

Opgave 18 (4 %)

Tegn hvordan nedenstående union-find datastruktur ser ud efter FIND(7) efterfulgt af FIND(3), når der anvendes stikomprimering.

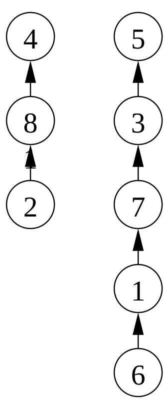


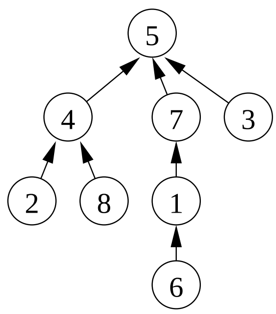


Svar: ______________________

Opgave 19 (4 %)

Tegn hvordan nedenstående union-find datastruktur ser ud efter UNION(2,7), når der anvendes både union-by-size og stikomprimering.





Svar: ______________________

Transitionssystem Counting
Configurations: $\{[i,j] \mid i,j \geq 1\}$
$[i,j] \triangleright [i,2j]$ **if** $j \leq 10$
$[i,j] \triangleright [i-1,j-10]$ **if** $j > 10 \wedge i > 1$

Opgave 20 (4 %)

For hver af nedenstående udsagn, angiv om de er en invariant for ovenstående transitionssystem Counting. Startkonfigurationen antages at være $[10,10]$.

	Ja	Nej
$1 \leq j \leq 10$	☐	☒
$1 \leq i \leq 10$	☒	☐
$10 \leq j \leq 20$	☒	☐
$j \geq i$	☒	☐
$i * 20 - j \leq 200$	☒	☐

Opgave 21 (4 %)

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

	Ja	Nej
$\mu(i,j) = i$	☐	☒
$\mu(i,j) = i - j$	☐	☒
$\mu(i,j) = i * 20 - j$	☒	☐
$\mu(i,j) = i^2 - j$	☐	☒
$\mu(i,j) = j - i$	☐	☒

(Opgavesættet fortsætter)

Algoritme Loop(n)
Inputbetingelse : heltal $n \geq 1$
Outputkrav : –
Metode : $i \leftarrow 1$;
$j \leftarrow n - 1$;
$\{I\}$ **while** $j > 0$ **do**
  $j \leftarrow j - i$;
  $i \leftarrow i + i$

Opgave 22 (4 %)

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

	Ja	Nej
$i + j = n$	☒	☐
$0 \leq i \leq n$	☐	☒
$j \leq n$	☒	☐
$0 \leq j \leq n$	☐	☒
$i = 2^j$	☐	☒

Opgave 23 (4 %)

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

	Ja	Nej
$\mu(i, j, n) = j$	☐	☒
$\mu(i, j, n) = n + j$	☒	☐
$\mu(i, j, n) = n - i$	☐	☒
$\mu(i, j, n) = j + i$	☐	☒
$\mu(i, j, n) = i * j$	☐	☒

Opgave 24 (4 %)

Nedenstående algoritme beregner $r = \lceil \log \log n \rceil$, dvs. det mindste r hvor $2^{2^r} \geq n$. For at vise gyldigheden af algoritmen skal I_n, I_r og I_s være invarianter omkring variablerne n, r og s. Angiv invarianter hvormed gyldigheden af algoritmen kan bevises (bevis for invarianterne kræves ikke).

Algoritme LogLog(n)
Inputbetingelse : heltal $n \geq 2$
Outputkrav : $r = \lceil \log \log n \rceil$
Metode : $s = 2$;
$r = 0$;
$\{I_n \wedge I_r \wedge I_s\}$ **while** $s < n$ **do**
$s \leftarrow s * s$;
$r \leftarrow r + 1$

Svar I_n: $n = n_0$

Svar I_r: $r \geq 0$

Svar I_s: $s = 2^{2^r} \quad \wedge \quad s < n^2$

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: $n^2 - s$

Opgave 25 (4 %)

Betragt en dobbelt-kædet liste hvor vi vedligeholder en pointer p til et element i listen. Operationerne next(p) og prev(p) flytter p til henholdsvis den umiddelbare forgænger og efterfølger til p i listen. Operationen reset(p) flytter p til det første element i listen ved gentagne gang at kalde prev(p), og tager tid $O(i)$ hvis når p peger på det $i$te element i listen.
For at argumentere at operationerne next, prev og reset tager $O(1)$ *amortiseret tid*, kræves en passende potentiale funktion. For hver af nedenstående funktioner, angiv om de kan anvendes som en potentiale funktion for et sådant argument.

	Ja	Nej
$\Phi(i) = i^2$	☐	☒
$\Phi(i) = i$	☒	☐
$\Phi(i) = \log i$	☐	☒
$\Phi(i) = 3i$	☒	☐
$\Phi(i) = 1$	☐	☒