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: Onsdag den 31. marts 2010, kl. 12.30-14.30
Eksamenslokale: Åbogade 34, Benjaminbygningen indgang B
Tilladte medbragte hjælpemidler: Alle sædvanlige hjælpemidler (lærebøger og notater). Computer må ikke medbringes.
Materiale der udleveres til eksaminanden:

Årskort ____________________

Navn ____

Skriftlig Eksamen
Algoritmer og Datastrukturer 1 (2003-ordning)

Datalogisk Institut
Aarhus Universitet

Onsdag den 31. marts 2010, kl. 12.30-14.30

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
$n + n^2$ er $O(n^2)$?	☐	☐
$n^3 \cdot \log n$ er $O(n^3)$?	☐	☐
$n^3 + \log n$ er $O(n^3)$?	☐	☐
$2^{\log n}$ er $O(n^3)$?	☐	☐
3^n er $O(\sqrt{n})$?	☐	☐

Opgave 2 (4 %)

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

$$4n^2$$
$$2^{3\log n}$$
$$2^n$$
$$1/\log n$$
$$\sqrt{n} \cdot \log n$$

Svar: ____

Opgave 3 (4 %)

I det følgende angiver f_i og g_i positive ikke-aftagende funktioner, hvor $f_1(n) = O(g_1(n))$ og $f_2(n) = O(g_2(n))$. Angiv hvilke af følgende udsagn der er sande.

	Ja	Nej
$f_1(n) + f_2(n) = O(g_1(n) + g_2(n))$?	☐	☐
$f_1(n) - f_2(n) = O(g_1(n) - g_2(n))$?	☐	☐
$f_1(n) * f_2(n) = O(g_1(n) * g_2(n))$?	☐	☐
$f_1(n) \;/\; f_2(n) = O(g_1(n) \;/\; g_2(n))$?	☐	☐
$1 \;/\; f_1(n) = \Omega(1 \;/\; g_1(n))$?	☐	☐

Opgave 4 (4 %)

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

Algoritme Loop1(n)
  $s = 1$
  for $i = 1$ **to** n
    for $j = 1$ **to** n
      $s = s + 1$

Algoritme Loop2(n)
  $s = 1$
  for $i = 1$ **to** n
    for $j = i$ **to** n
      for $k = i$ **to** j
        $s = s + 1$

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

Svar Loop1: ____________________

Svar Loop2: ____________________

Svar Loop3: ____________________

Opgave 5 (4 %)

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

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

Algoritme Loop2(n)
  $i = 1$
  while $i \leq n$
    $i = i + i$

Algoritme Loop3(n)
  $i = 2$
  while $i \leq n$
    $i = i * i$

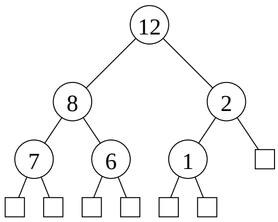
Svar Loop1: ____________________

Svar Loop2: ____________________

Svar Loop3: ____________________

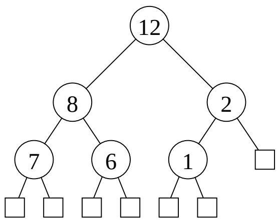
Opgave 6 (4 %)

Tegn hvordan nedenstående binære max-heap ser ud efter indsættelse af elementet 4.



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

Svar: ____

Opgave 9 (4 %)

Betragt radix-sort anvendt på nedenstående liste af binære tal ($d = 6$, $k = 2$). Angiv den delvist sorterede liste efter at radix-sort har sorteret tallene efter de *tre* mindst betydende cifre.

110101 110001 000101 000001 101101 001001

Svar: ____

Opgave 10 (4 %)

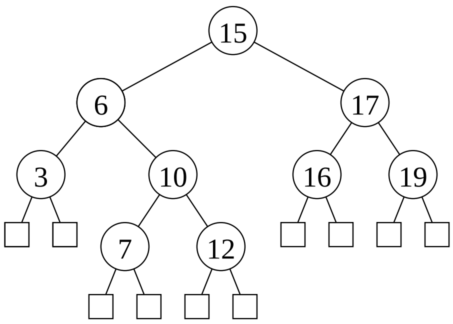
Angiv resultatet af at anvende PARTITION(A,11,19) på nedenstående array.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	8	16	1	6	2	4	13	17	15	3	18	5	9	11	24	12	14	10	7	22

Svar: ____

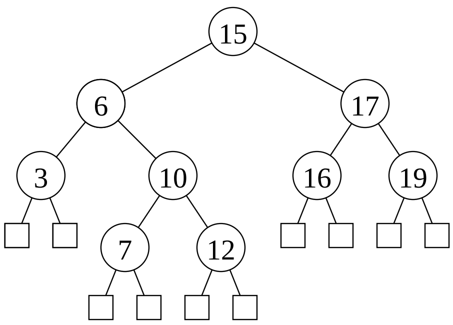
Opgave 11 (4 %)

Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter indsættelse af elementet 11.



Svar: ________________________

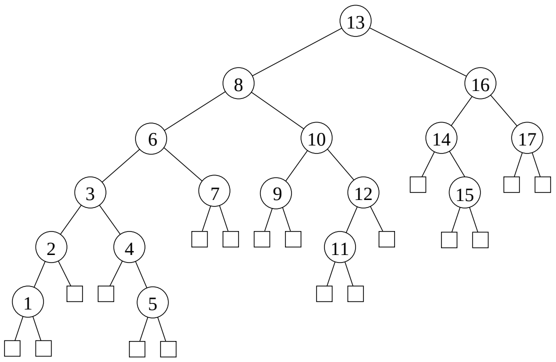
Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter slettelse af elementet 15.



Svar: ________________________

Opgave 12 (4 %)

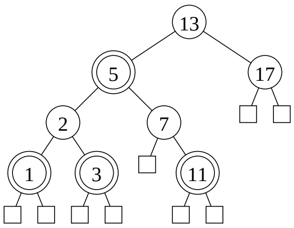
Angiv hvorledes knuderne i nedenstående binære søgetræ kan farves røde og sorte, således at det resulterende træ er et lovligt rød-sort træ.



Svar: ______________________________

Opgave 13 (4 %)

Tegn hvordan nedenstående rød-sorte træ (dobbeltcirkler angiver røde knuder) ser ud efter indsættelse af elementet 4.



Svar: ______________________________

Opgave 14 (4 %)

Tegn en hashtabel hvor der anvendes kædede lister til at håndtere kollisioner, når hashfunktionen er $h(k) = 2k \bmod 5$ og der indsættes elementerne 5, 7, 2, 4, 8, 3, og 12 i den givne rækkefølge.

Svar: ______________________________________

Opgave 15 (4 %)

Tegn hvordan en hashtabel der anvender *linear probing* ser ud efter at elementerne 2, 6, 12, 1, 5, 10, 16, og 0 indsættes i den givne rækkefølge, når hashfunktionen er $h(k) = 2k \bmod 11$.

0	1	2	3	4	5	6	7	8	9	10

Svar: ____

Opgave 16 (4 %)

Tegn hvordan en hashtabel der anvender *dobbelt hashing* ser ud efter at elementerne 11, 4, 3, 5, og 12 indsættes i den givne rækkefølge, når hashfunktionerne er $h_1(k) = k \bmod 8$ og $h_2(k) = 1 + 2k \bmod 8$, og hashtabellen har størrelse 8.

0	1	2	3	4	5	6	7

Svar: ____

Opgave 17 (4 %)

Angiv for hver af nedenstående datastrukturer indeholdende n elementer, hvor lang tid det tager at finde predecessoren til et element x, dvs. det største element $\leq x$ i datastrukturen, som funktion af n i O-notation.

Rød-sort søgetræ ? Svar: ________________

Binær max-heap ? Svar: ________________

Hashtabel med linear probing ved 50% fyldningsgrad ? Svar: ________________

Opgave 18 (4 %)

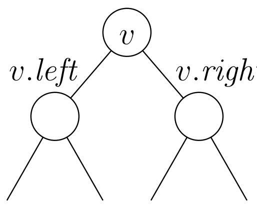
For en sorteret liste af tal $x_1, \ldots, x_n$, definerer vi

$$\mathrm{ssq}(x_1, \ldots, x_n) = (x_n - x_{n-1})^2 + (x_{n-1} - x_{n-2})^2 + \cdots + (x_3 - x_2)^2 + (x_2 - x_1)^2$$

F.eks. er

$$\mathrm{ssq}(1, 3, 8, 11, 13, 19) = (19 - 13)^2 + (13 - 11)^2 + (11 - 8)^2 + (8 - 3)^2 + (3 - 1)^2 = 78$$

Betragt et søgetræ hvor hver knude v ud over et element $v.e$, gemmer summen $v.ssq$ som er lig ssq(elementerne i v's undertræ), og $v.min$ og $v.max$ som er hhv. det mindste element og største element i v's undertræ. 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). F.eks. betegner $v.right.min$ det mindste element i v's højre undertræ.



Svar $v.min$ = ____________________

Svar $v.max$ = ____________________

Svar $v.ssq$ = ____________________

Opgave 19 (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)
union(a,b)
union(c,d)
union(a,c)
union(a,e) Svar: ____________________

Transitionssystem Frem-og-tilbage
Konfigurationer: $\{[i,j,k] \mid \text{heltal } i,j,k \wedge i \geq 0 \wedge j \geq 0 \wedge k \geq 0\}$

$$
\begin{array}{lll}
[i,j,k] \triangleright [i-1, j+2, k] & \textbf{if} & i \geq 1 \\
[i,j,k] \triangleright [i, j-1, k+3] & \textbf{if} & j \geq 1 \\
[i,j,k] \triangleright [i+1, j, k-7] & \textbf{if} & k \geq 7
\end{array}
$$

Opgave 20 (4 %)

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

	Ja	Nej
$i+j+k \geq 0$	☐	☐
$i \leq j$	☐	☐
$6i+3j+k \leq 10n$	☐	☐
$2i+j \leq 3k$	☐	☐
$i-1 = j+2$	☐	☐

Opgave 21 (4 %)

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

	Ja	Nej
$\mu(i,j,k) = i+j+k$	☐	☐
$\mu(i,j,k) = k$	☐	☐
$\mu(i,j,k) = 52i+25j+8k$	☐	☐
$\mu(i,j,k) = 10i+4j+k$	☐	☐
$\mu(i,j,k) = 6i+3j+k$	☐	☐

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

Opgave 22 (4 %)

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

	Ja	Nej
$i \geq 1$	☐	☐
$r = i!$	☐	☐
$i! \cdot r! = n!$	☐	☐
$r = n!/i!$	☐	☐
$r = n!$	☐	☐

Opgave 23 (4 %)

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

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

Opgave 24 (4 %)

Givet et positivt heltal n, beregner nedenstående algoritme heltalslogaritmen af n. Denne værdi betegnes

$$\log(n) = \max\{i \mid i \text{ heltal} \land 2^i \le n\} \ .$$

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

Algoritme IntegerLog(n)
Inputbetingelse : positivt heltal n
Outputkrav : $r = \log(n)$
Metode : $p \leftarrow 1;$
$r \leftarrow 0;$
$\{I_p \ \land \ I_r\}$ **while** $2 * p \le n$ **do**
    $p \leftarrow 2 * p;$
    $r \leftarrow r + 1;$

Svar I_p: ______________________________

Svar I_r: ______________________________

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: ______________________________

Opgave 25 (4 %)

Antag en binær tæller implementeres som et (uendeligt) array af bits $B[0]B[1]B[2]\cdots$. Det antages at tælleren forøges med én vha. følgende metode.

Algoritme inc
Metode : $i \leftarrow 0;$
while $B[i] = 1$ **do**
    $B[i] \leftarrow 0;$
    $i \leftarrow i + 1;$
$B[i] \leftarrow 1$

For at argumentere at inc tager amortiseret $O(1)$ tid kræves en potentiale funktion. Angiv en potentiale funktion hvorved tiden kan bevises. Argumentation for tiden kræves ikke.

Svar $\Phi(B[0]B[1]B[2]\cdots)$: ______________________________