

DATALOGISK INSTITUT, AARHUS UNIVERSITET

Det Naturvidenskabelige Fakultet
EKSAMEN
Grundkurser i Datalogi
<b>Algoritmer og Datastrukturer 1 (2003-ordning)</b>
Antal sider i opgavesættet (incl. forsiden): 12 (tolv)
Eksamensdag: Onsdag den 30. marts 2011, kl. 13.30-15.30
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 30. marts 2011, kl. 13.30-15.30

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:

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

**Opgave 1 (4%)**

	Ja	Nej
$n^2 + 2^n$ er $O(n^4)$ ?	<input type="checkbox"/>	<input type="checkbox"/>
$7n^2 + 3n + 4$ er $O(n^3)$ ?	<input type="checkbox"/>	<input type="checkbox"/>
$\log n + \log^2 n$ er $O(\sqrt{n})$ ?	<input type="checkbox"/>	<input type="checkbox"/>
$3n^2 + 2n^3$ er $O(n)$ ?	<input type="checkbox"/>	<input type="checkbox"/>
$\sqrt{7}$ er $O(\log n)$ ?	<input type="checkbox"/>	<input type="checkbox"/>

**Opgave 2 (4%)**

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

$$\begin{aligned} &(\sqrt{n})^5 \\ &7n^2 \\ &(\log n)^3 \\ &2^{3 \log n} \\ &\log(n^3) \end{aligned}$$

Svar: \_\_\_\_\_

**Opgave 3 (4%)**

Betragt RANDOMIZED-SELECT anvendt til at finde det  $\lceil \sqrt{n} \rceil$  mindste element i et array med  $n$  elementer. Angiv for hvert af nedenstående spørgsmål svaret som funktion af  $n$  i  $O$ -notation.

Worst-case antal rekursive kald? Svar: \_\_\_\_\_

Forventede antal rekursive kald? Svar: \_\_\_\_\_

Worst-case tid? Svar: \_\_\_\_\_

Forventet tid? Svar: \_\_\_\_\_

#### Opgave 4 (4%)

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

**Algoritme Loop1( $n$ )**

```
s = 1
for i = 1 to n
  for j = 1 to n + 1 - i
    s = s + 1
```

**Algoritme Loop2( $n$ )**

```
s = 0
for i = 1 to n
  while s ≤ i
    s = s + 1
```

**Algoritme Loop3( $n$ )**

```
s = 1
for i = 1 to n
  while s ≤ i
    s = 2 * s
```

Svar Loop1: \_\_\_\_\_

Svar Loop2: \_\_\_\_\_

Svar Loop3: \_\_\_\_\_

#### Opgave 5 (4%)

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

**Algoritme Loop1( $n$ )**

```
i = 1
j = 1
while i ≤ n
  while j ≤ i
    j = 2 * j
  i = 2 * i
```

**Algoritme Loop2( $n$ )**

```
i = 1
while i < n
  i = i * n
```

**Algoritme Loop3( $n$ )**

```
i = 1
j = 1
while j ≤ n
  j = j + i
  i = i + 1
k = 1
while k ≤ i
  k = 2 * k
```

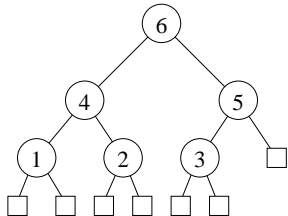
Svar Loop1: \_\_\_\_\_

Svar Loop2: \_\_\_\_\_

Svar Loop3: \_\_\_\_\_

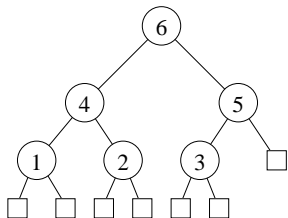
**Opgave 6 (4%)**

Tegn hvordan nedenstående binære max-heap ser ud efter indsættelse af elementet 7.



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 7, 5, 2, 9, 3, 0 og 6 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	10	4	3	7	2	9	6	8	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.

3311    3412    2433    4410    1212    1133

Svar: \_\_\_\_\_

**Opgave 10 (4%)**

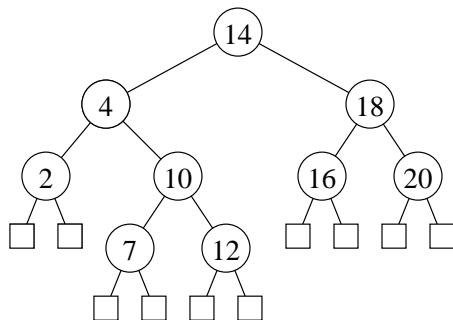
Angiv resultatet af at anvende PARTITION( $A,6,14$ ) på nedenstående array.

	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: \_\_\_\_\_

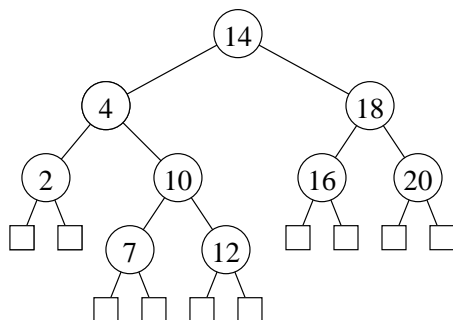
**Opgave 11 (4%)**

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



Svar: \_\_\_\_\_

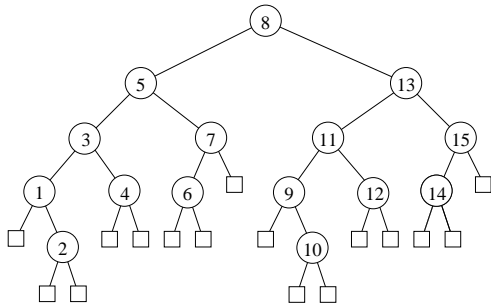
Tegn hvordan nedenstående ubalancerede binære søgetræ ser ud efter slettelse af elementet 4.



Svar: \_\_\_\_\_

**Opgave 12 (4%)**

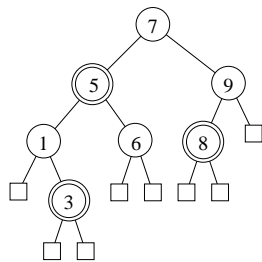
Angiv hvorledes knuderne i nedenstående binære søgetræ kan farves røde og sorte, således at det resulterende træ er et lovligt rød-sort træ.



Svar: \_\_\_\_\_

**Opgave 13 (4%)**

Tegn hvordan nedenstående rød-sort træ (dobbeltcirkler angiver røde knuder) ser ud efter indsættelse af elementet 2.



Svar: \_\_\_\_\_

**Opgave 14 (4%)**

Tegn en hashtabel hvor der anvendes kædede lister til at håndtere kollisioner, når hash-funktionen er  $h(k) = 5k \bmod 6$  og der indsættes elementerne 1, 2, 3, 4, 6, 9 og 12 i den givne rækkefølge.

Svar: \_\_\_\_\_

**Opgave 15 (4%)**

Tegn hvordan en hashtabel der anvender *linear probing* ser ud efter at elementerne 1, 2, 15, 5, 6, 3, 11 og 8 indsættes i den givne rækkefølge, når hashfunktionen er  $h(k) = 3k \text{ mod } 10$ .

0	1	2	3	4	5	6	7	8	9

Svar: \_\_\_\_\_

**Opgave 16 (4%)**

Tegn hvordan en hashtabel der anvender *quadratic probing* ser ud efter at elementerne 1, 7, 14 og 23 indsættes i den givne rækkefølge, når hashfunktionen er

$$h(k, i) = 3k + 3i + 2i^2 \text{ mod } 13$$

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

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(*b*,*d*)  
union(*a*,*f*)

Svar: \_\_\_\_\_



**Opgave 18 (4%)**

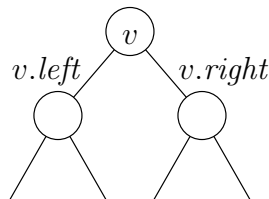
For en sorteret liste af tal  $x_1, \dots, x_n$ , definerer vi

$$\text{maxgap}(x_1, \dots, x_n) = \max\{x_n - x_{n-1}, x_{n-1} - x_{n-2}, \dots, x_3 - x_2, x_2 - x_1\}$$

F.eks. er

$$\text{maxgap}(7, 12, 19, 23) = \max\{23 - 19, 19 - 12, 12 - 7\} = 7$$

Betragt et søgetræ hvor hver knude  $v$  ud over et element  $v.e$ , gemmer  $v.\text{maxgap}$  som er lig  $\text{maxgap}$ (elementerne i  $v$ 's undertræ), og  $v.\text{min}$  og  $v.\text{max}$  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  $v.\text{left}$  og  $v.\text{right}$  (det kan antages at disse begge eksisterer). F.eks. betegner  $v.\text{right}.\text{min}$  det mindste element i  $v$ 's højre undertræ.



Svar  $v.\text{min}$  = \_\_\_\_\_

Svar  $v.\text{max}$  = \_\_\_\_\_

Svar  $v.\text{maxgap}$  = \_\_\_\_\_

**Opgave 19 (4%)**

Betragt RADIX-SORT anvendt til at sortere  $n$  binære tal med 128 bits hver. Angiv for  $n = 1.000.000$  og  $n = 1.000$  det antal bits  $r$  hvert kald af COUNTING-SORT skal kigge på for at minimere udførelstiden for RADIX-SORT.

$n = 1.000.000$  ? Svar  $r =$  \_\_\_\_\_

$n = 1.000$  ? Svar  $r =$  \_\_\_\_\_

**Transitionssystem** Nedtælling  
Konfigurationer:  $\{[i, k] \mid \text{heltal } i, k \wedge i \geq 1 \wedge k \geq 0\}$   
 $[i, k] \triangleright [i - 1, k] \quad \text{if } i \geq 2 \wedge i \text{ ulige}$   
 $[i, k] \triangleright [i/2, k + 1] \quad \text{if } i \geq 2 \wedge i \text{ lige}$

**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  $n \geq 1$ .

	Ja	Nej
$i + k \leq n$	<input type="checkbox"/>	<input type="checkbox"/>
$\lfloor \log i \rfloor + k = \lfloor \log n \rfloor$	<input type="checkbox"/>	<input type="checkbox"/>
$k \leq \lfloor \log n \rfloor$	<input type="checkbox"/>	<input type="checkbox"/>
$i - k \geq 0$	<input type="checkbox"/>	<input type="checkbox"/>
$2^k \leq i$	<input type="checkbox"/>	<input type="checkbox"/>

**Opgave 21 (4%)**

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

	Ja	Nej
$\mu(i, k) = i$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(i, k) = \log i$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(i, k) = \lceil \log i \rceil$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(i, k) = i + k$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(i, k) = 2i + k$	<input type="checkbox"/>	<input type="checkbox"/>

**Algoritme** SumOfSquares( $n$ )

Inputbetingelse : heltal  $n \geq 1$

Outputkrav :  $r = \sum_{i=1}^n i^2 = 1^2 + 2^2 + 3^2 + \dots + n^2$

Metode :  $j \leftarrow 1$

$r \leftarrow 1$

$\{I\}$  **while**  $j < n$  **do**

$j \leftarrow j + 1$

$r \leftarrow r + j * j$

**Opgave 22** (4%)

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

	Ja	Nej
$j \leq n$	<input type="checkbox"/>	<input type="checkbox"/>
$r = j^2$	<input type="checkbox"/>	<input type="checkbox"/>
$r^2 \leq n^2$	<input type="checkbox"/>	<input type="checkbox"/>
$r = j(j + 1)/2$	<input type="checkbox"/>	<input type="checkbox"/>
$r \leq n^2$	<input type="checkbox"/>	<input type="checkbox"/>

**Opgave 23** (4%)

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

	Ja	Nej
$\mu(j, r) = j$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(j, r) = n - j$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(j, r) = n^2/r$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(j, r) = n^2 - r$	<input type="checkbox"/>	<input type="checkbox"/>
$\mu(j, r) = (n - j)^2$	<input type="checkbox"/>	<input type="checkbox"/>

**Opgave 24 (4%)**

Givet et positivt heltal  $n$ , så beregner nedenstående algoritme summen

$$\sum_{j=1}^n j^3 = \frac{n^2(n+1)^2}{4}$$

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).

**Algoritme** Sum( $n$ )

Inputbetingelse : positivt heltal  $n \geq 1$

Outputkrav :  $r = \sum_{j=1}^n j^3$

Metode :  $i \leftarrow n - 1$ ;

$r \leftarrow n * n * n$ ;

$\{I_i \wedge I_r\}$  **while**  $i \geq 1$  **do**

$r \leftarrow r + i * i * i$ ;

$i \leftarrow i - 1$ ;

Svar  $I_i$ : \_\_\_\_\_

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  $\mu$ : \_\_\_\_\_

**Opgave 25 (4%)**

Betragt en kø implementeret ved hjælp af to stakke  $head$  and  $tail$ , som hver understøtter push og pop i worst-case  $O(1)$  tid. For en tom kø er  $tail = head = \emptyset$ . Enqueue og Dequeue implementeres ved hjælp af følgende metoder.

**Algoritme** Enqueue( $x$ )

Metode : push( $tail$ ,  $x$ )

**Algoritme** Dequeue()

Inputbetingelse :  $|head| + |tail| \geq 1$

Metode : **if**  $|head| = 0$  **then**

**while**  $|tail| > 0$  **do**

push( $head$ , pop( $tail$ ))

**return** pop( $head$ )

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

Svar  $\Phi(head, tail)$  : \_\_\_\_\_