

---

**Algoritmi și structuri de date (I). Seminar 5:** Stabilirea ordinului de complexitate pornind de la numărul de operații efectuate. Proprietăți și calcule cu ordine de complexitate. Analiza complexității în cazul mediu.

---

**Problema 1** Să se determine ordinul de creștere al valorii variabilei  $x$  (în funcție de  $n$ ) după execuția algoritmilor **alg1**, **alg2**, **alg3**, **alg4**, **alg5** și **alg6**.

<b>alg1(integer <math>n</math>)</b>	<b>alg2(integer <math>n</math>)</b>
1: $x \leftarrow 0$	1: $x \leftarrow 0$
2: <b>for</b> $i \leftarrow 1, n$ <b>do</b>	2: $i \leftarrow n$
3: <b>for</b> $j \leftarrow 1, i$ <b>do</b>	3: <b>while</b> $i \geq 1$ <b>do</b>
4: <b>for</b> $k \leftarrow 1, j$ <b>do</b>	4: $x \leftarrow x + 1$
5: $x \leftarrow x + 1$	5: $i \leftarrow i \text{DIV} 2$
6: <b>end for</b>	6: <b>end while</b>
7: <b>end for</b>	7: <b>return</b> $x$
8: <b>end for</b>	
9: <b>return</b> $x$	

<b>alg3(integer <math>n</math>)</b>	<b>alg4(integer <math>n</math>)</b>
1: $x \leftarrow 0$	1: $x \leftarrow 0$
2: $i \leftarrow n$	2: $i \leftarrow n$
3: <b>while</b> $i \geq 1$ <b>do</b>	3: <b>while</b> $i \geq 1$ <b>do</b>
4: <b>for</b> $j \leftarrow 1, n$ <b>do</b>	4: <b>for</b> $j \leftarrow 1, i$ <b>do</b>
5: $x \leftarrow x + 1$	5: $x \leftarrow x + 1$
6: <b>end for</b>	6: <b>end for</b>
7: $i \leftarrow i \text{DIV} 2$	7: $i \leftarrow i \text{DIV} 2$
8: <b>end while</b>	8: <b>end while</b>
9: <b>return</b> $x$	9: <b>return</b> $x$

*Indicație.* **alg1**:

$$x = \sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^j 1 = \sum_{i=1}^n \sum_{j=1}^i j = \sum_{i=1}^n \frac{i(i+1)}{2} = \frac{n(n+1)(n+2)}{6} \in \Theta(n^3)$$

**alg2**:  $x$  va conține numărul de cifre ale reprezentării în baza 2 a lui  $n$ , adică  $\lfloor \lg_2(n) \rfloor + 1 \in \Theta(\lg n)$ .

**alg3**: Spre deosebire de exemplul anterior, pentru fiecare valoare a lui  $i$  variabila  $x$  este mărită cu  $n$ , deci valoarea finală va fi  $n(\lfloor \lg_2(n) \rfloor + 1) \in \Theta(n \lg n)$ .

**alg4**: Este similar algoritmului **alg3** însă pe linia 4 limita superioară a ciclului **for** este  $i$  în loc de  $n$ . Variabila  $x$  va conține suma  $n + \lfloor \frac{n}{2} \rfloor + \dots + \lfloor \frac{n}{2^k} \rfloor$  cu  $k$  având proprietatea că  $2^k \leq n < 2^{k+1}$ . Ordinul de mărime al lui  $x$  se poate stabili pornind de la observația că  $2^{k+1} - 1 \leq x < 2(2^{k+1} - 1)$ . Deci  $x \in \Theta(2^k) = \Theta(n)$ .

<b>alg5(integer <math>n</math>)</b>	<b>alg6(integer <math>n</math>)</b>
1: $x \leftarrow 0$	1: $x \leftarrow 0$
2: $i \leftarrow 1$	2: $i \leftarrow 2$
3: <b>while</b> $i < n$ <b>do</b>	3: <b>while</b> $i < n$ <b>do</b>
4: $x \leftarrow x + 1$	4: $x \leftarrow x + 1$
5: $i \leftarrow 2 * i$	5: $i \leftarrow i * i$
6: <b>end while</b>	6: <b>end while</b>
7: <b>return</b> $x$	7: <b>return</b> $x$

**alg5:** Variabila  $x$  va conține cel mai mare număr natural  $k$  cu proprietatea că  $2^k \leq n$ , deci  $x = \lfloor \lg_2 n \rfloor + 1 \in \Theta(\lg n)$ .

**alg6:** Variabila  $x$  va conține cel mai mare număr natural  $k$  cu proprietatea că  $2^{2^k} \leq n$  deci  $x = \lfloor \lg_2 \lg_2 n \rfloor + 1 \in \Theta(\lg_2 \lg_2 n)$ .

**Problema 2** Să se determine termenul dominant și ordinul de creștere pentru expresiile:

- (a)  $2\lg n + 4n + 3n\lg n$
- (b)  $2 + 4 + 6 + \dots + 2n$
- (c)  $\frac{(n+1)(n+3)}{n+2}$
- (d)  $2 + 4 + 8 + \dots + 2^n$

*Indicație.* (a) Termenul dominant în  $2\lg n + 4n + 3n\lg n$  este evident  $3n\lg n$  iar ordinul de creștere este  $n\lg n$ .  
 (b) Termenul dominant al sumei  $2 + 4 + 6 + \dots + 2n$  nu este  $2n$  ci  $n^2$  întrucât  $2 + 4 + 6 + \dots + 2n = n(n+1)$ . Deci ordinul de creștere este chiar  $n^2$ .

(c) Termenul dominant este  $n^2/(n+2)$  iar ordinul de creștere este  $n$ .

(d) Întrucât  $2 + 4 + 8 + \dots + 2^n = 2(1 + 2 + 4 + \dots + 2^{n-1}) = 2(2^n - 1)$ , termenul dominant este  $2 \cdot 2^n$  iar ordinul de creștere este  $2^n$ .

**Problema 3** Să se arate că:

- (a)  $n! \in \mathcal{O}(n^n)$ ,  $n! \in \Omega(2^n)$
- (b)  $\lg n! \in \Theta(n\lg n)$
- (c)  $2^n \in \mathcal{O}(n!)$
- (d)  $\sum_{i=1}^n i\lg i \in \mathcal{O}(n^2\lg n)$
- (e)  $\lg(n^k + c) \in \Theta(\lg n)$ ,  $k > 0, c > 0$
- (f)  $\sum_{i=1}^n i^k \in \Theta(n^{k+1})$

*Indicație.* (a) Întrucât  $n! \leq n^n$  pentru orice  $n$  natural, rezultă imediat că  $n! \in \mathcal{O}(n^n)$ . Pe de altă parte,  $n! \geq 2^{n-1}$  pentru orice  $n$ , deci  $n! \in \Omega(2^n)$ .

(b) Se pornește de la aproximația Stirling  $n! \simeq \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$  care este adevărată pentru valori mari ale lui  $n$ . Mai exact, există  $c_1, c_2 \in \mathbb{R}_+$  și  $n_0 \in \mathbb{N}$  astfel încât

$$\sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \frac{c_1}{n}\right) \leq n! \leq \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \frac{c_2}{n}\right)$$

pentru orice  $n \geq n_0$ .

Prin logaritmare se obține:

$$\ln(2\pi n)/2 + \ln(1 + c_1/n) + n \ln n - n \leq \ln n! \leq \ln(2\pi n)/2 + \ln(1 + c_2/n) + n \ln n - n$$

deci  $\ln n! \in \Theta(n \ln n)$ . Facând abstracție de baza logaritmului se obține  $\lg n! \in \Theta(n\lg n)$

(c) Cum  $2^n < n!$  pentru orice  $n \geq 4$  rezultă că  $2^n \in \mathcal{O}(n!)$ .

(d)  $\sum_{i=1}^n i\lg i \leq \lg n \sum_{i=1}^n i \leq n^2 \lg n$ , deci  $\sum_{i=1}^n i\lg i \in \mathcal{O}(n^2 \lg n)$ .

(e) Pentru valori suficient de mari ale lui  $n$  are loc  $\lg n^k \leq \lg(n^k + c) \leq \lg(2n^k)$ , adică  $k\lg n \leq \lg(n^k + c) \leq k\lg n + \lg 2$ , deci  $\lg(n^k + c) \in \Theta(\lg n)$ .

**Problema 4** Care dintre următoarele afirmații sunt adevărate ? Demonstrați.

- (a)  $\sum_{i=1}^n i^2 \in \Theta(n^2)$ ,  $\sum_{i=1}^n i^2 \in \mathcal{O}(n^2)$ ,  $\sum_{i=1}^n i^2 \in \Omega(n^2)$
- (b)  $cf(n) \in \Theta(f(n))$ ,  $f(cn) \in \Theta(f(n))$
- (c)  $2^{n+1} \in \mathcal{O}(2^n)$ ,  $2^{2n} \in \mathcal{O}(2^n)$  ?

*Indicație.* (a) Întrucât  $\sum_{i=1}^n i^2 = n(n+1)(2n+1)/6$  rezultă că  $\sum_{i=1}^n i^2 \in \Theta(n^3)$ , deci  $\sum_{i=1}^n i^2 \in \Omega(n^2)$  însa celelalte două afirmații sunt false.

(b) Întrucât  $c_1 f(n) \leq c f(n) \leq c_2 f(n)$  pentru constante  $c_1$  și  $c_2$  satisfăcând  $0 < c_1 \leq c \leq c_2$  rezultă că  $c f(n) \in \Theta(f(n))$ . În schimb  $f(cn) \in \Theta(f(n))$  nu este adevărată pentru orice funcție  $f$  și orice constantă  $c$ . De exemplu  $f(n) = \exp(n)$  și  $c > 1$  nu satisfac această proprietate întrucât nu există  $c'$  astfel încât  $\exp(cn) \leq c' \exp(n)$  pentru valori mari ale lui  $n$ .

(c) Întrucât  $2^{n+1} = 2 \cdot 2^n$  rezultă că  $2^{n+1} \in \Theta(2^n)$  deci implicit și  $2^{n+1} \in \mathcal{O}(2^n)$ . Pe de altă parte,  $\lim_{n \rightarrow \infty} 2^{2n}/2^n = \infty$  prin urmare  $2^{2n} \notin \mathcal{O}(2^n)$  dar  $2^{2n} \in \Omega(2^n)$ .

**Problema 5** Propuneți un algoritm pentru determinarea celor mai mici două valori dintr-un tablou. Determinați numărul de comparații efectuate asupra elementelor tabloului și stabiliți ordinul de complexitate al algoritmului.

**Problema 6** Propuneți un algoritm de complexitate liniară pentru a determina tabloul frecvențelor cumulate pornind de la tabloul frecvențelor simple. Pentru un tablou  $(f_1, \dots, f_n)$  de frecvențe, tabloul frecvențelor cumulate  $(fc_1, \dots, fc_n)$  se caracterizează prin  $fc_i = \sum_{j=1}^i f_j$ .

**Problema 7** Să se stabilească ordinul de complexitate pentru următorii algoritmi ce prelucrează date de volum  $n$ .

```

(a)  for  $i \leftarrow 1, n$  do
      ...  $\Theta(1)$ 
      for  $j \leftarrow 1, 2i$  do
        ...  $\Theta(1)$ 
         $k \leftarrow j$   $\Theta(1)$ 
        while  $k \geq 0$  do
          ...  $\Theta(1)$ 
           $k \leftarrow k - 1$ 
        endwhile
      endfor
    endfor

(b)  h  $\leftarrow 1$ 
      while  $h \leq n$  do
        ...  $\Theta(1)$ 
         $h \leftarrow 2 * h$ 
      endwhile

(c)  calcul  $(x[0..n-1])$ 
       $k \leftarrow 0$ 
      for  $i \leftarrow 0, n-1$  do
        for  $j \leftarrow 0, n-1$  do
           $y[k] \leftarrow x[i] * x[j]$ 
           $k \leftarrow k + 1$ 
        endfor
      endfor return  $y[1..k]$ 

```

**Problema 8** Propuneți un algoritm de complexitate  $\Theta(n^2)$  și unul de complexitate  $\Theta(n)$  pentru evaluarea unui polinom de grad  $n$ .

### Probleme suplimentare/teme

1. Să se determine ordinul de mărime al variabilei  $x$  după execuția următoarelor prelucrări (și implicit ordinul de complexitate a algoritmului) :

---

```

1:   $x \leftarrow 0$ 
2:   $j \leftarrow n$ 
3:  while  $j \geq 1$  do
4:    for  $i \leftarrow 1, j$  do
5:       $x \leftarrow x + 1$ 
6:    endfor
7:     $j \leftarrow j \text{DIV} 3$ 
8:  endwhile

```

---

2. Se consideră un text constituit din  $N$  cuvinte, lungimea maximă a unui cuvânt fiind  $M$ . Asupra textului se aplică următoarea prelucrare:

Pas 1. Se ordonează crescător literele din fiecare cuvânt

Pas 2. Se ordonează lexicografic cuvintele din text

De exemplu, textul "exemplu de transformare a textului" va suferi următoarele transformări:

Pas 1. "eelmpux ed aaefmnorrst a eilttuux"

Pas 2. "a aaefmnorrst ed eelmpux eilttuux"

Stabiliți ordinul de complexitate al algoritmului în ipoteza că: (a) se utilizează un algoritm de sortare de complexitate pătratică,  $O(n^2)$ ; (b) se utilizează un algoritm de sortare de complexitate  $O(n \lg n)$ .

3. Se consideră următorul algoritm:

```

1: alg(int  $n$ )
2:  $d \leftarrow 2$ 
3: while  $d * d < n$  do
4:   if  $n \text{ MOD } d == 0$  then
5:     return False
6:   end if
7:    $d \leftarrow d + 1$ 
8: end while
9: return True

```

Care este ordinul de complexitate al algoritmului?

4. Se consideră următorul algoritm:

```

1: alg(int  $n$ , int  $k$ )
2:  $p \leftarrow 1$ 
3: while  $k > 0$  do
4:   if  $k \text{ MOD } 2 == 1$  then
5:      $p \leftarrow p * n$ ;  $k \leftarrow k - 1$ 
6:   else
7:      $n \leftarrow n * n$ ;  $k \leftarrow k \text{ DIV } 2$ 
8:   end if
9: end while
10: return  $p$ 

```

Care este cel mai favorabil caz? Care este cel mai defavorabil caz? Care este ordinul de complexitate a algoritmului?

5. Se consideră următorul algoritm:

```

1: alg(int  $n$ )
2:  $i \leftarrow n/2$ ;  $nr \leftarrow 0$ 
3: while  $i \leq n$  do
4:    $j \leftarrow n$ 

```

```
5:  while  $j \geq 1$  do  
6:     $nr \leftarrow nr + 1$   
7:     $j \leftarrow j/2$   
8:  end while  
9:   $i \leftarrow i + 1$   
10: end while  
11: return  $nr$ 
```

Ce returnează algoritmul? Care este ordinul de complexitate?