Complexitatea Algoritmilor
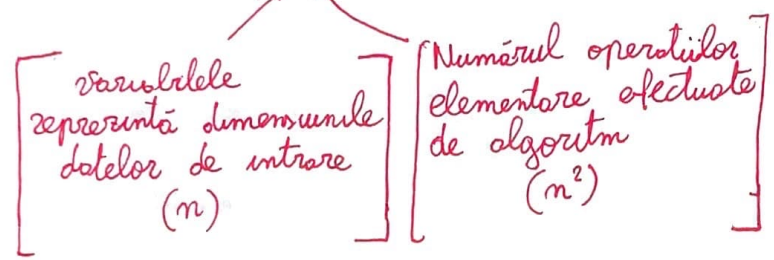
Algoritmul este MECANIC } → cuvânt cheie

COMPLEXITATE
- → COMPUTAȚIONALĂ → timp de execuție → Depinde de:
  - Sistemul de operare
  - Computatorul
  - Programe în fundal
- → SPAȚIALĂ → memoria utilizată
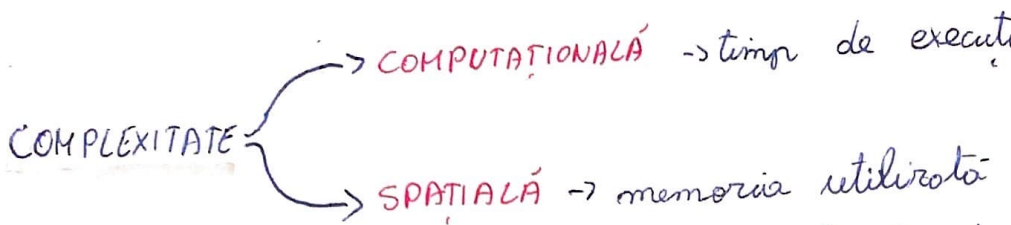
COMPLEXITATEA COMPUTAȚIONALĂ: număr de operații elementare efectuate de algoritm în raport de dimensiunea datelor de intrare.

NOTAȚIE ASINTOTICĂ: $\mathcal{O}(n^2)$ $(n \in \mathbb{N}^*, n \to \infty)$

[variabilele reprezintă dimensiunile datelor de intrare (n)] [Numărul operațiilor elementare efectuate de algoritm (n^2)]

$$\underbrace{\mathcal{O}(n^2)}_{\text{mai bun}} < \mathcal{O}(n^3)$$

- - - - - - - -

OPERAȚIILE ELEMENTARE

- OPERAȚII ARITMETICE + OP. ATRIBUIRE $(+, -, \cdot, /)$
- OPERAȚII DE DECIZIE ("if") + OPERAȚIA DE SALT ("goto")
- OPERAȚII DE CITIRE/SCRIERE

REGULI DE SIMPLIFICARE A EXPRESIILOR:

a) dintr-o expresie se păstrează termenul dominant $(n \to \infty)$

$$\mathcal{O}(n^3 + n^2) \tilde{\Rightarrow} \mathcal{O}(n^3)$$
$$\mathcal{O}(n^2 + 2^n) \tilde{\Rightarrow} \mathcal{O}(2^n)$$

b) constantele nu contează

$$\mathcal{O}(5n^2 + 3) \tilde{\Rightarrow} \mathcal{O}(n^2) \to \text{Complexitate maximă (caz defavorabil :)}$$

$$\left\{ \begin{array}{l} \Omega\ (n^2) \\ \Theta\ (n^2) \end{array} \right\} \to \text{complexitate medie}$$

Ex. Maximul dintr-un vector

de la "foc"

```
int v[100], n, i, max;
scanf("%d", n);              → citire = 1 op
for(i=0; i<n; i++) scanf("%d", &v[i]);   → citire = n operații
max = v[0];                  → atribuire = 1 op          depinde de buclă
for(i=1; i<n; i++)           → instr. atomecă
    if(v[i] > max)
        max = v[i];          → atribuire = de cel mult n-1 ori operații
printf("%d", max);           → scriere = 1 operație
```

$$\left[1 + n + 1 + n + 1 = 2n + 3 \Rightarrow \mathcal{O}(n) \right]$$

Ex. sortarea prin interschimbare

```
int v[100], n, i, max, J, aux;
scanf ("%d", n) -> citire = 1 operație
For (i=0; i<n; i++)    scanf ("%d", v[i]); -> citire = n operații
For (i=0; i<n; i++)
    For (J=i+1; J<n; J++)
        if (v[i] > v[J]) {
            aux = v[i];
            v[i] = v[J];
            v[J] = aux;
        }
    }
For (i=0; i<n; i++)    printf ("%d", v[i]); -> scriere = n operații
```

de cel mult $3\left(\dfrac{n \cdot (n-1)}{2}\right)$

$$1 + n + \frac{3n(n-1)}{2} + n = \frac{3n^2 + n + 2}{2}$$

$$\to O\left(\frac{3n^2+n+2}{2}\right) \to O(3n^2+n+2) = O(n^2)$$

$i=0 \Rightarrow J \in \{1, \ldots, n-1\} \Rightarrow n-1$ ori

$i=1 \Rightarrow J \in \{2, \ldots, n-1\} \Rightarrow n-2$ ori

⋮

$i=n-2 \Rightarrow J \in \{n-1\} \Rightarrow 1$ ori

$$(n-1)(n-2) \cdots (1) =$$

$$= \frac{(n-1) \cdot (n-1+1)}{2} = \boxed{\frac{n(n-1)}{2}}$$

ORICE ALGORITM PRESUPUNE CITIREA UNOR DATE DE INPUT ȘI SCRIEREA UNOR DATE DE OUTPUT; COMPLEXITATEA NU POATE FI MAI MICĂ DECÂT COMPLEXITATEA DATELOR DE INPUT ȘI DATELOR DE IEȘIRE.

CARE ESTE COMPLEXITATEA MINIMĂ CARE CALCULEAZĂ SUMĂ ELEMENTELOR DIAGONALEI PRINCIPALE DINTR-O MATRICE PATRATICA DE DIM. N.

Var. 1
```
Sum = 0;
for (i=0; i<n; i++)
    for (J=0; J<n; J++)
        if (i==J)
            sum = sum + a[i][J]
```
$O(n^2)$

Var 2
```
Sum = 0;
for (i=0; i<n; i++)
    sum = sum + a[i][i];
```
$O(n)$

$O(n^2)$
```
for (i=0; i<n; i++)
    for (J=0; J<n; J++)
        scanf ("%d", a[i][J]);
```