Mulțimi

- Reflexivă $\quad a \mathcal{R} a \quad \forall a \in S$
- Simetrică $\quad a \mathcal{R} b = b \mathcal{R} a$
- Antisim.
- Tranzitivă

---

ceil ↗ $\lceil \; \rceil$ approx. floor ↘ $\lfloor \; \rfloor$

Sume și Recurențe

Gauss; $\sum_{i=1}^{n} i^2 = \dfrac{2n^3 + 3n^2 + n}{}$

factorial: $\begin{cases} n! = (n-1)! \cdot n, & n \geq 1 \\ 1! = 0! = 1 \end{cases}$

Fibonacci: $\begin{cases} f(n) = f(n-1) + f(n-2) \\ f(1) = f(0) = 1 \end{cases}$

```
unsigned long factorial ( uint n ){
if (n==0)
    return 1;
else
    return n* fact(n-1);
}
```

$T(0) = c$ (const. timp)

$T(1) = c + T(0) = 2c$

$T(2) = c + T(1) = 3c$

$\_\ \_\ \_\ \_\ \_$

$T(n) = (n+1)c$

Tehnici demonstrare

- dem. directă (deducție)
- Contradicție
- Inducție

p.p. a $\exists$ lui MAX

dacă $MAX \in \mathbb{N} \to MAX+1 \in \mathbb{N}$

$MAX + 1 > MAX \Rightarrow \nexists$ lim

Inducția

- Caz inițial $P(k_0)$ adev.
- Pasul de inducție $P(k) \Rightarrow P(k+1)$

Caz de bază → stop. la rec.

```
if (n == 0)
    return 0
```

Pas de ind. → ret. $n * f(n-1)$

Interval (Rec)

```
void f ( int start, int end) {
    if ( start == end)     caz bază
        ...
    else
        ...
        f( start , end - 1)    pas ind.
```

Algoritmi

→ Metodă pt. a obține un rez. dorit

Propr. → G.

→

→

Timp Rulare / Mem. Consumată

Analiza Algoritmilor

for

for

$$c \cdot n \cdot n$$

$$T(n) = c\,n^2$$

3 cazuri

- Cel mai favorabil
- Cel mai nefavorabil
- Average case

Notații asimptotice

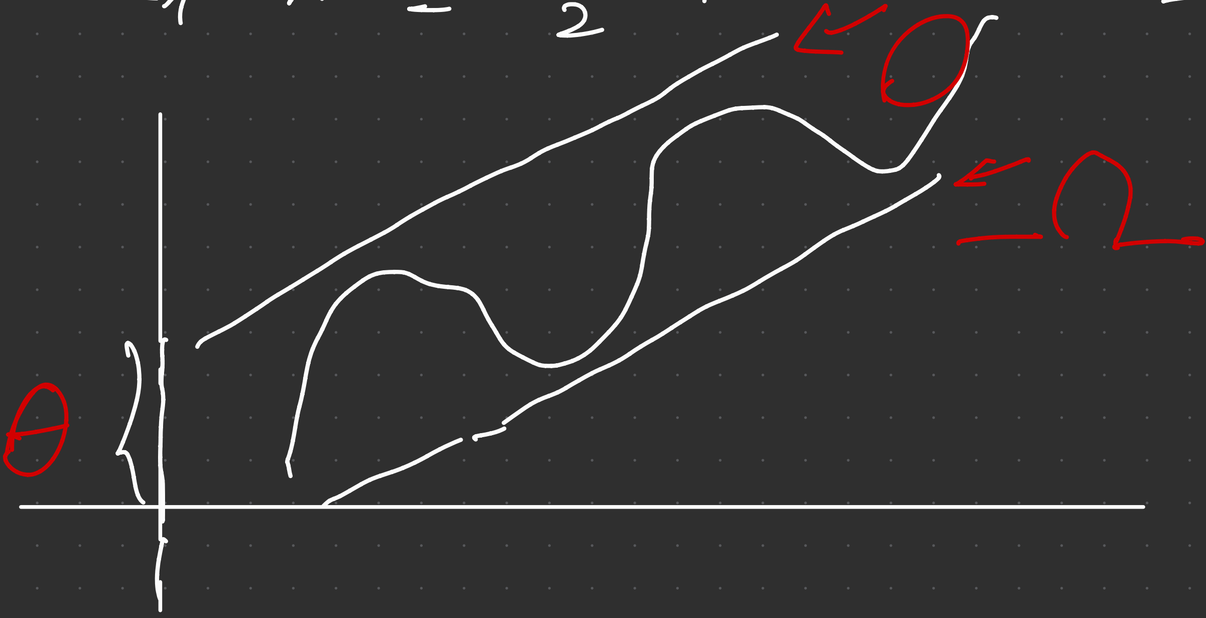
fct. $g(n) \to \Theta(g(n))$ (theta)

$\Theta(g(n)) = \{ f(n) ; \exists$ const > 0 c_1, c_2 și n_0 a.î.

$0 \le c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n)$
$\forall n \ge n_0 \}$

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

$$c_1 \cdot n^2 \le \frac{1}{2} n^2 - 3n \le c_2 n^2$$



Aprecierea t. execuție

pt. t. constant $\Theta(1) \Rightarrow O(1)$

$(O(n^0))$

Nu $\exists$ alte constante în $O(\ldots)$

for (. . . .) { nc

a[ ] ...

for(i=0; i<100; i++)

$a[i] = i+1$

$$c \cdot 100 = c \longrightarrow O(1)$$

---

for(i=0; i<n; i++)

for(j=0, j<i, j++)

$a[j] = j+1$

$$O(n^2)$$

$$\underbrace{c + c + \dots + c}_{n} = nc$$

for(k=1 ; k<=n ; k *=2)

log n ori

Căutarea binară

$$\underline{T(1)=1}$$

$$T(n) = T(n/2) + 1$$

$$T(n) = \log n \Rightarrow \Theta(n)$$

Profilarea Algoritmilor

→ VS Profiling (CPU, MEM)