= COMPLEXITĂȚI =

1) "O": SPUNEM CĂ $f \in O(g)$ DACĂ $(\exists)$ c, n_0 (CONSTANTE) >0 a.î. $(\forall)$ $n \geq n_0$ AVEM $f(n) \leq c \cdot g(n)$. (FUNCȚIA SĂ FIE MĂRGINITĂ SUPERIOR DE LA UN ANUMIT RANG ÎNCOLO).

Ex: $3n^3 \in O(n^3)$; $n^2 \in O(n^3)$; $\log(n)+3 \in O(n)$

$\boxed{O: \text{"}\leq\text{"}}$

2) "Ω": SPUNEM CĂ $f \in \Omega(g)$ DACĂ $(\exists)$ c, n_0 (CONST.) >0 a.î. $(\forall)$ $n \geq n_0$ AVEM $f(n) \geq c \cdot g(n)$. (MĂRGINITĂ INFERIOR)

Ex: $n^3 \in \Omega(n^3)$; $\frac{n^3}{2} \in \Omega(n^3)$; $n^2 \in \Omega(n^3)$; $n^2-4n+17 \notin \Omega(n^3)$

$\boxed{\Omega: \text{"}\geq\text{"}}$

3) "θ": SPUNEM CĂ $f \in \theta(g)$ DACĂ $(\exists)$ $c1, c2, n_0$ (CONST.) >0 a.î. $(\forall)$ $n \geq n_0$ AVEM $c1 \cdot g(n) \leq f(n) \leq c2 \cdot g(n)$. $\boxed{\theta: \text{"}=\text{"}}$

Ex: $n^2 - 2n \cdot \log(0,5n) \in \theta(n^2)$

4) "o": SPUNEM CĂ $f \in o(g)$ DACĂ $(\forall)$ $c>0$, $(\exists)$ n_0 a.î. $(\forall)$ $n \geq n_0$ AVEM $f(n) < c \cdot g(n)$. $\boxed{o: \text{"}<\text{"}}$ → MĂRGINITĂ STRICT SUPERIOR

Ex: $n^3 \notin o(n^3)$ $\quad n^2 \in o(n^3)$

$\frac{n^3}{2} \notin o(n^3)$

5) "ω": SPUNEM CĂ $f \in \omega(g)$ DACĂ $(\forall)$ $c>0$, $(\exists)$ n_0 a.î. $n \geq n_0$ AVEM $f(n) > c \cdot g(n)$. $\boxed{\omega: \text{"}>\text{"}}$ → MĂRGINITĂ STRICT INFERIOR

Ex: $n^3 \notin \omega(n^3)$; $n^4 \notin \omega(n^4)$; $n^4 \in \omega(n^3)$; $n \cdot \log n \in \omega(n)$

$\begin{cases} O(g): \text{"}\leq\text{"} \\ o(g): \text{"}<\text{"} \end{cases}$ $\quad \theta(g): \text{"}=\text{"} \quad$ $\begin{cases} \Omega(g): \text{"}\geq\text{"} \\ \omega(g): \text{"}>\text{"} \end{cases}$

* ÎN $\Omega(g)$ SE AFLĂ TOATE FUNCȚIILE CU TERMENUL DOMINANT "$\geq$" CEL DIN "g"
* ÎN $O(g)$ " ——— " ——— " " $\leq$ " " — " —
* ÎN $\theta(g)$ " ——— " ——— " " $=$ " " — " —

EX:

$O(n^3)$	$\theta(n^3)$	$\Omega(n^3)$
n^2	n^3	n^4
$3n^2 + n \cdot \log(n^4)$	$3n^3 + n + \log n$	$2^n - n^4$
$11n$	$n^3 + \log(n^{99})$	$4n^3 + 4n^2$
$n\sqrt{n}$		$n^3\sqrt{n}$

CRESTERE ASIMPTOTICĂ:

$$1 < \log n < n < n \cdot \log n < n^2 < n^2 \log n < \dots < 2^n$$

= RECURENTE =

ÎN GENERAL, PT. UN ALGORITM RECURSIV, AVEM URMĂTORUL TIMP DE RULARE:

$$T(n) = T(\text{SUBPROBLEMĂ } 1) + T(\text{" — " — " } 2) + \vdots + T(\text{" — " — " } n) + \underbrace{\Delta(n)}_{\text{TIMP DIVIZIUNE}} + \underbrace{C(n)}_{\text{TIMP COMBINARE}}$$

! $T(n) = 2T\left(\frac{n}{2}\right) + 1 \Rightarrow O(n)$

! $T(n) = T\left(\frac{n}{a}\right) + T\left(\frac{(a-1)n}{a}\right) + O(n) \rightarrow O(n \cdot \log n)$; $T(n) = T\left(\frac{n}{b}\right) + 1 \Rightarrow \Theta(n \cdot \log n)$

MERGE SORT: $T(n) = 2T\left(\frac{n}{2}\right) + O(n) \rightarrow O(n \cdot \log n)$

INSERTION SORT: $T(n) = T(n-1) + n \rightarrow O(n^2)$

CĂUTARE BINARĂ: $T(n) = T\left(\frac{n}{2}\right) + 1 \rightarrow O(\log n)$

= TEOREMA MASTER =

APLIC PE RECURENTE DE FORMA $\boxed{T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)}$.

1) DACĂ $f(n) \in O(n^{\log_b a - \varepsilon})$, $\varepsilon > 0 \Rightarrow T(n) = \Theta(n^{\log_b a})$

2) DACĂ $f(n) \in \Theta(n^{\log_b a}) \Rightarrow T(n) = \Theta(n^{\log_b a} \cdot \log_2 n)$

3) DACĂ $f(n) \in \Omega(n^{\log_b a + \varepsilon})$, pt $\varepsilon > 0$, $\Rightarrow T(n) \in \Theta(f(n))$

! $1 + \frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{n} \approx \ln n$

EXERCIȚII:

1) $n^3 \in\ ? \Rightarrow n^3 \in O(n^3)$

$g(n) = n^3$

$f \in O(g) \Leftrightarrow (\exists)\ c, n_0 > 0$ a.î. $(\forall)\ n \geq n_0,\ f(n) \leq c \cdot g(n)$

ALEGEM $c = 1;\ n_0 = 1 \Rightarrow n^3 \leq 1 \cdot n^3$ (A), $(\forall)\ n \geq 1$

2) $200n^3 \in O(n^3)$

$g(n) = n^3;\ f(n) = 200n^3$

$f \in O(g) \Leftrightarrow (\exists)\ c, n_0 > 0$ a.î. $(\forall)\ n \geq n_0,\ f(n) \leq c \cdot g(n)$

ALEGEM $c = 201,\ n_0 = 1 \Rightarrow 200n^3 \leq 201 \cdot n^3$ (A), $(\forall)\ n \geq 1$

*3) $n \cdot \log_2 n \in \Theta(\log_2 n!) \Leftrightarrow \log_2 n! \in \Theta(n \cdot \log_2 n)$

$f(n) = n \cdot \log_2 n$

$g(n) = \log_2 n!$

$f \in \Theta(g) \Leftrightarrow (\exists)\ c_1, c_2, n_0 > 0$ a.î. $(\forall)\ n \geq n_0,\ c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$

a) $c_1 \cdot g(n) \leq f(n)$

$c_1 \cdot \log_2 n! \leq n \cdot \log_2 n = \log_2 n^n$

Alegem $c_1 = 1,\ n_0 = 1 \Rightarrow \log_2 n! \leq \log_2 n^n \Leftrightarrow n! \leq n^n$ (A)

DEM. PRIN INDUCȚIE;

$n = 1 \Rightarrow 1! = 1 \leq 1^1$ (A)

$P(k) \rightarrow P(k+1)$

$P(k):\ k! \leq k^k$ A

$P(k+1):\ (k+1)! \leq (k+1)^{k+1}$

$k!\cancel{(k+1)} \leq (k+1)^k \cdot \cancel{(k+1)}$

$k! \leq (k+1)^k$ (A) $(k! \leq k^k)$

b) $f(n) \leq c_2 \cdot g(n)$

$\log_2 n^n \leq c_2 \cdot \log_2 n!$

$\log_2 n! = \log_2 1 + \dots + \log_2 n \geq \log_2 \frac{n}{2} + \dots + \log_2 n \geq \frac{n}{2} \log_2(n/2) =$

$= \frac{n}{2}(\log_2 n - 1) \geq c_2 \cdot n \cdot \log_2 n$

$c_2 = \frac{1}{4};\ n_0 = 4$

4) $f(n)+g(n)\in\Theta(\max\{f(n);g(n)\})$

$(\exists)\ c_1, c_2, n_0>0$ a.î. $(\forall)\ n\geq n_0 \Rightarrow c_1\cdot g(n)\leq f(n)\leq c_2\cdot g(n)$ → CAZ GENERA

a) $\left.\begin{array}{l} c_1\cdot\max\{f(n);g(n)\}\leq f(n)+g(n) \\ c_1=1;\ n_0=1 \end{array}\right\}\Rightarrow A$

b) $\left.\begin{array}{l} f(n)+g(n)\leq c_2\cdot\max\{f(n),g(n)\} \\ c_2=2;\ n_0=1 \end{array}\right\}\Rightarrow A$

5) $\left.\begin{array}{l} f\in O(g) \\ g\in O(h) \end{array}\right\}\Rightarrow f\in O(h)$

$\left.\begin{array}{l} f\in O(g)\Leftrightarrow(\exists)\ c_1,n_0>0 \text{ a.î. } (\forall)\ n\geq n_0,\ f(n)\leq c_1\cdot g(n) \\ g\in O(h)\Leftrightarrow(\exists)\ c_2,n_0'>0 \text{ '' ---- '' ---- '' };\ g(n)\leq c_2\cdot h(n) \end{array}\right\}\Rightarrow$

$\left.\begin{array}{l} \Rightarrow f(n)\leq c_1\cdot g(n)\leq c_1\cdot c_2\cdot h(n)\Rightarrow f(n)\leq c_1\cdot c_2\cdot h(n) \\ f\in O(h)\Leftrightarrow(\exists)\ c,n_0>0 \text{ a.î. } (\forall)\ n\geq n_0,\ f(n)\leq c\cdot h(n) \end{array}\right\}\Rightarrow f\in O(h)$

6) $o(f(n))\cap\omega(f(n))=\emptyset$

$g\in o(f(n))\Leftrightarrow(\forall)\ c_1>0,\ (\exists)\ n_0$ a.î. $(\forall)\ n>n_0;\ g(n)<c_1\cdot f(n)$

$g\in\omega(f(n))\Leftrightarrow(\forall)\ c_2>0,\ (\exists)$ '' ---- '' ---- ''; $g(n)>c_2\cdot f(n)$

PRESUPUN PRIN ABSURD CĂ $g(n)\in(o(f(n))\cap\omega(f(n)))\Rightarrow$

$\Rightarrow\begin{cases} g(n)\in o(f(n))\Rightarrow g(n)<c_1 f(n) \\ g(n)\in\omega(f(n))\Rightarrow g(n)>c_2 f(n) \end{cases}\ (\forall)\ c_1,c_2$

Aleg $c_1=c_2=1\Rightarrow\begin{cases} g(n)<f(n) \\ g(n)>f(n) \end{cases};\ (\forall)\ n>n_0;n_0'$

Aleg $n''>\max(n_0 \text{ și } n_0')\Rightarrow\left.\begin{cases} g(n'')<f(n'') \\ g(n'')>f(n'') \end{cases}\right\}\Rightarrow$ ∡∡

7) $(n+a)^b \in \Theta(n^b)$; $a>0$; $b>1$

$f\in\Theta(g) \Rightarrow (\exists)\, c_1; c_2; n_0>0$ a.î. $(\forall)\, n>n_0 \Rightarrow c_1\cdot g(n) \le f(n) \le c_2\cdot g(n)$

a) $c_1 \cdot n^b \le (n+a)^b$

$\left.\begin{array}{l} c_1 \cdot n^b \le n^b + C_b^1 n^{b-1} a + \dots + C_b^b a^b \\ c_1 = 1 \text{ și } n_0 = 1 \end{array}\right\} \Rightarrow (A) \Rightarrow 0 \le C_b^1 n^{b-1}a+\dots+C_b^b a^b$

b) $(n+a)^b \le c_2 \cdot n^b$

$(n+a)^b = n^b + C_b^1 n^{b-1}a+\dots+C_b^b a^b \le n^b(1+C_b^1 a+\dots+C_b^b a^b) \le$

$\le n^b\cdot a^b(1+C_b^1+C_b^2+\dots+C_b^b) = n^b\cdot a^b\cdot 2^b(b+1)$

$c_2 = a^b\cdot 2^b(b+1)$; $n_0=1 \Rightarrow A$

8) $f(n)+o(f(n)) \in \Theta(f(n))$

$f\in\Theta(g) \Leftrightarrow (\exists)\, c_1, c_2, n_0>0$ a.î. $(\forall)\, n>n_0 \Rightarrow c_1\cdot g(n) \le f(n) \le c_2\cdot g(n)$

a) $c_1\cdot f(n) \le f_n + o(f(n))$

$c_1=1 \Rightarrow A$, $(\forall)\, n>n_0$

b) $f(n)+o(f(n)) \le c_2\cdot f(n)$

$g\in o(f(n)) \Rightarrow (\forall)\, c>0, (\exists)\, n_0$ a.î. $(\forall)\, n>n_0 \Rightarrow g(n) < c\cdot f(n)$

$\left.\begin{array}{l} f(n)+o(f(n)) < f(n)+c\cdot f(n) \le \underset{c_2=2}{2}\cdot f(n) \end{array}\right\} \Rightarrow A$

---

1) $T(n)=T\left(\frac{n}{a}\right)+T\left(\frac{(a-1)n}{a}\right) + \overset{O(n)}{n} \in O(n\cdot\log n)$ → MERGESORT

$T(n)=T\left(\frac{n}{3}\right)+T\left(\frac{2n}{3}\right)+n$

PRESUPUN CĂ $T\left(\frac n3\right) \le c\cdot\frac n3\cdot\log_2\frac n3$

TB. SĂ ARĂTĂM CĂ $T(n)\le c\cdot n\cdot\log_2 n$

$T(n)=T\left(\frac n3\right)+T\left(\frac{2n}{3}\right)+n \le c\cdot\frac n3\cdot\log_2\frac n3 + 2c\cdot\frac n3\cdot\log_2\frac{2n}{3}+n =$

$= c\cdot\frac n3\left(\log_2 n-\log_2 3+2\cdot(\underbrace{\log_2 2}_{1}+\log_2 n-\log_2 3)\right)+n =$

$= c\cdot\frac n3\left(3\cdot\log_2 n-3\cdot\log_2 3+2\right)+n = cn\left(\log_2 n-\log_2 3+\frac23\right)+n =$

$= c\cdot n\cdot\log_2 n - cn\underbrace{\left(\log_2 3-\frac23-\frac1c\right)}_{>0} \le c\cdot n\cdot\log_2 n$ ✓

2) $T(n) = 2T\left(\frac{n}{2}\right) + 1 \in O(n)$

PRESUPUN $T\left(\frac{n}{2}\right) \leq c \cdot \frac{n}{2} - b$

TB. SĂ DEM. $T(n) \leq c \cdot n - b$

$T(n) = 2T\left(\frac{n}{2}\right) + 1 \leq \cancel{2}c \cdot \frac{n}{\cancel{2}} - 2b + 1 = cn - 2b + 1 \leq cn - b \left.\begin{matrix} \\ \end{matrix}\right\} \Rightarrow ✓$

$b > 1$

3) $T(n) = T(n/2) + n \in O(n)$

PRESUPUN $T(n/2) \leq c \cdot \frac{n}{2}$

TB. SĂ DEM. $T(n) \leq c \cdot n$

$T(n) = T(n/2) + n \leq c \cdot \frac{n}{2} + n = n\left(1 + \frac{c}{2}\right) \leq c \cdot n,\ (\forall)\ c \geq 2$

4) $T(n) = T(n-1) + 1 \in O(n)$

PRESUPUN $T(n-1) \leq c(n-1)$

TB. SĂ DEM. $T(n) \leq c \cdot n$

$T(n) = T(n-1) + 1 \leq c(n-1) + 1 = cn - (c-1) \leq cn,\ (\forall)\ c \geq 1$

5) $T(n) = T(n-1) + \overset{O(n)}{n} \in O(n^2)$ → INSERTION SORT

PRESUPUN $T(n-1) \leq c \cdot (n-1)^2$

TB. SĂ DEM $T(n) \leq c \cdot n^2$

$T(n) = T(n-1) + n \leq c(n-1)^2 + n = c(n^2 - 2n + 1) + n = \cancel{c \cdot n^2} - 2 \cdot c \cdot n + c + n \leq \cancel{c n^2}$

$-2 \cdot c \cdot n + c + n \leq 0 \Rightarrow 2 \cdot c \cdot n - c - n \geq 0$

$n\left(2c - 1 - \frac{c}{n}\right) \geq 0$, (A), $(\forall)\ c \geq 1$

6) $T(n) = T\left(\frac{n}{2}\right) + 1 \in O(\log n)$

PRESUPUN $T\left(\frac{n}{2}\right) \leq c \cdot \log_2 \frac{n}{2}$

TB. SĂ DEM: $T(n) \leq c \cdot \log_2 n$

$T(n) = T\left(\frac{n}{2}\right) + 1 \leq c \cdot \log_2 \frac{n}{2} + 1 = c(\log_2 n - \overbrace{\log_2 2}^{1}) + 1 = c \cdot \log_2 n - (c-1) \leq$

$\leq c \cdot \log_2 n,\ (\forall)\ c \geq 1$

7) $T(n) = 2T(n-1) + 1 \in O(2^n)$

PRESUPUN $T(n-1) \leq c \cdot 2^{n-1} - b$

TB. SĂ DEM $T(n) \leq c \cdot 2^n - b$

$T(n) = 2T(n-1) + 1 \leq 2 \cdot c \cdot 2^{n-1} - 2b + 1 = c \cdot 2^n - 2b + 1 \leq c \cdot 2^n - b,\ (\forall)\ b \geq 1$

= SORTĂRI =

$O(n^2)$: BUBBLE / INSERTION / SELECTION SORT

$O(n \cdot \log n)$: MERGE / QUICK / HEAP SORT

$O(n)$: COUNT / RADIX SORT

1) BUBBLESORT:

5 1 4 2 8 9 (LUĂM GRUPURI DE 2 ELEMENTE)

1<5 => INTERSCHIMB

1 5 4 2 8 9

4<5 => SWAP

1 4 5 2 8 9

2<5 => SWAP

1 4 2 5 8 9

PARCURGI LISTA PÂNĂ CÂND E ORDONATĂ. IEI "ok=0" CA SĂ VERIFICI DACĂ S-AU PRODUS INTERSCHIMBĂRI. ok=1 => STOP.

1 2 4 5 8 9 → SORTAT

2) INSERTION SORT:

SORTAT | ZONA NESORTATĂ

7 | 8 5 2 4 6 3

7<8 → NU SE INTERSCHIMBĂ

7 8 | 5 2 4 6 3

8>5; 7>5

5 7 8 | 2 4 6 3

2<8; 2<7; 2<5

2 5 7 8 | 4 6 3

4<8; 4<7; 4<5; 4>2

2 4 5 7 8 | 6 3

6<8; 6<7; 6>5

2 4 5 6 7 8 | 3

3<8 .... 3<4; 3>2

2 3 4 5 6 7 8 → SORTAT

3) SELECTION SORT:

SORTAT | NESORTAT

| (64) 25 12 22 11

SWAP

PARCURGEM VECTORUL ȘI GĂSIM MINIMUL.

11 | (25) 12 22 64

11 12 | (25) 22 64

11 12 22 | (25) 64

11 12 22 25 64 → SORTAT

---

4) MERGESORT: (DIVIDE ET IMPERA) ⇒ $T(n) = 2T(n/2) + O(n) \in O(n \cdot \log n)$

38 27 43 3 | 9 82 10

38 27 | 43 3   9 82 | 10

38 | 27   43 | 3   9 | 82   10

38 > 27   43 > 3   9 < 82   10

DIVIDE

27 38   3 43   9 82   10

3 27 38 43   9 10 82

(27 > 3; 27 < 43; 38 < 43)   (9 < 10; 82 > 10)

3 9 10 27 38 43 82 ⇒ SORTAT

(3 < 9; 27 > 9; 27 > 10; 27 < 82 ....)

MERGE

5) QUICKSORT: $\Rightarrow T(n) = T\left(\frac{n}{a}\right) + T\left(\frac{(a-1)n}{a}\right) + O(n) \in O(n \cdot \log n)$

$\Downarrow$

DIVIDE ET IMPERA

PIVOT (TOATE EL. DIN STÂNGA $\Rightarrow$ "$\leq$"; DIN DREAPTA $\Rightarrow$ "$>$")

5 11 22 7 4 900 100 1 32 $\boxed{13}$

$i=0$, $i=1$, $i=2$; $j=0$

$5<13$; $11<13$; $22 \not< 13 \to$ SWAP (22;13)

5 11 $\boxed{13}$ 7 4 900 100 1 32 22

$j=2$, $j=1$

$13<32$; $13 \not< 1 \to$ SWAP (13;1)

5 11 1 7 4 900 100 $\boxed{13}$ 32 22

$i=3$, $i=4$, $i=5$

$7<13$; $4<13$; $900 \not< 13 \to$ SWAP (900, 13)

5 11 1 7 $\boxed{4}$ | $\boxed{13}$ | 100 900 32 $\boxed{22}$ $(13<100 \checkmark)$

$j=3$

APLICĂM ACEIAȘI PAȘI PT PARTEA STÂNGĂ ȘI APOI PT. PARTEA DR..

5 11 1 7 $\boxed{4}$

$i=0$; $j=0$

$5 \not< 4 \to$ SWAP

$\boxed{4}$ 11 1 7 5

$j=2$, $j=1$

$4<7$; $4 \not< 1 \to$ SWAP

1 11 $\boxed{4}$ 7 5

$i=1$

$11 \not< 4 \to$ SWAP

$\boxed{1}$ | $\boxed{4}$ | 11 7 $\boxed{5}$

SORTAT

$i=0$; $j=0$

$\boxed{5}$ | 7 $\boxed{11}$

100 900 32 $\boxed{22}$

$i=0$; $j=0$

$100 \not< 22 \to$ SWAP

$\boxed{22}$ | 900 32 $\boxed{100}$

$j=2$, $j=1$

$22<32$; $22<900$

900 32 $\boxed{100}$

$i=0$; $j=0$

$900 \not< 100 \to$ SWAP

$\boxed{100}$ 32 900

$j=1$

$100 \not< 32 \to$ SWAP

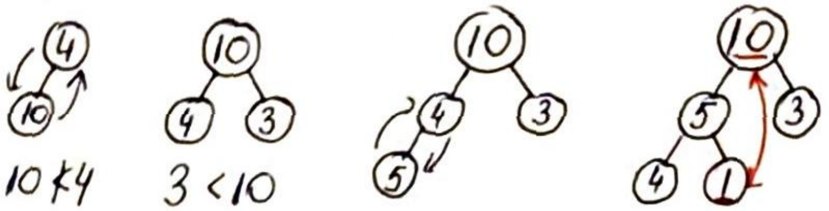
32 | $\boxed{100}$ | 900

$\Downarrow$

1 4 5 7 11 13 22 32 100 900 $\Rightarrow$ SORTAT

6) HEAPSORT: => $\in O(n \cdot \log n)$

CREĂM HEAP CU ELEMENTELE DATE: 4 10 3 5 1 (MAX HEAP)

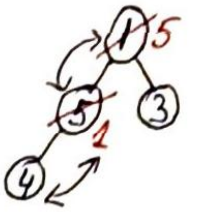


$10 \nless 4$ $3 < 10$ $5 \nless 4 ; 5 < 10$ $1 < 5$

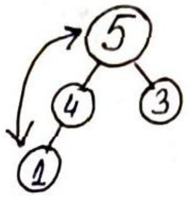
VECTORUL: 10 5 3 4 1

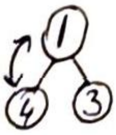
SWAP PRIMUL ȘI ULTIMUL ELEMENT DIN HEAP ȘI ELIMINĂM ULTIMUL ELEMEN

VECTOR: 1 5 3 4 | 10 VECTOR: 5 4 3 1 | 10

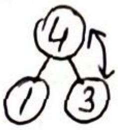


(REARANJĂM HEAP) =>





VECTOR: 1 4 3 | 5 10 =>



V: 4 1 3 | 5 10

VECTOR: 3 1 | 4 5 10 =>

V: 1 | 3 4 5 10

=> V: 1 3 4 5 10 => SORTAT

7) COUNTING SORT: $\Rightarrow O(n) \Rightarrow$ DACĂ AM UN ȘIR DE "n" ELEMENTE ÎN CARE FIECARE ELEMENT ARE MAX. "K" APARIȚII $\Rightarrow O(n+K)$

INPUT: 1 4 1 2 7 5 2 $\Rightarrow$ CHEILE $\in [0,9] \cap \mathbb{N}$

INDEX:	0	1	2	3	4	5	6	7	8	9	VECTOR FRECVENȚĂ
	0	2	2	0	1	1	0	1	0	0	$\Rightarrow$ DE CÂTE ORI APAR CHEILE
		0+2	2+2	4+0	4+1	5+1	6+0	6+1	7+0	7+0	

INDEX:	0	$\boxed{1}$	2	3	4	5	6	7	8	9	
	0	$\boxed{\cancel{2}\ 1}$	4	4	5	6	6	7	7	7	$\Rightarrow$ SUMA APARIȚII

$\hookrightarrow$ 1 apare la poziția 2; 2-1=1

AVEM 7 ELEMENTE ÎN INPUT $\Rightarrow$ VECTOR DE POZIȚII CU 7 ELEMENTE.

POZIȚII:	1	2	3	4	5	6	7	
	1	1	2	2	4	6	7	$\Rightarrow$ SORTAT

INPUT: 1 4 1 2 7 5 2 (PARCURGEM INPUT PAS CU PAS ȘI FACEM MODIFICĂRI ÎN "INDEX")

POZ: 2
2-1 = 1

POZ: 5 $\Rightarrow$ 5-1=4

POZ: 1
1-1=0

2 la poz: 4
4-1=3

NU E OPTIM PENTRU VALORI MARI.
SE FOLOSEȘTE ÎMPREUNĂ CU RADIXSORT.

```
for(i=0;i<n;i++)
    frecvența[v[i]]++;
for(i=1;i<=EL_MAX;i++)
    frecvența[i]+=frecvența[i-1];
for(i=0;i<n;i++)
{ output[frecvența[v[i]]-1]=v[i];
    frecvența[v[i]]--;
}
```

8) RADIX SORT: ⇒ $O(n)$ $(O(n+k))$

INPUT: 170 45 75 90 802 24 2 66

ÎNCEPEM DE LA CEL MAI NESEMNIFICATIV BIT (CIFRA UNITĂȚILOR).

LE SORTĂM FOLOSIND COUNTING SORT. (DACĂ (∃) CIFRE EGALE, SE PĂSTREAZĂ ORDINEA ORIGINALĂ.

170 90 802 2 24 45 75 66

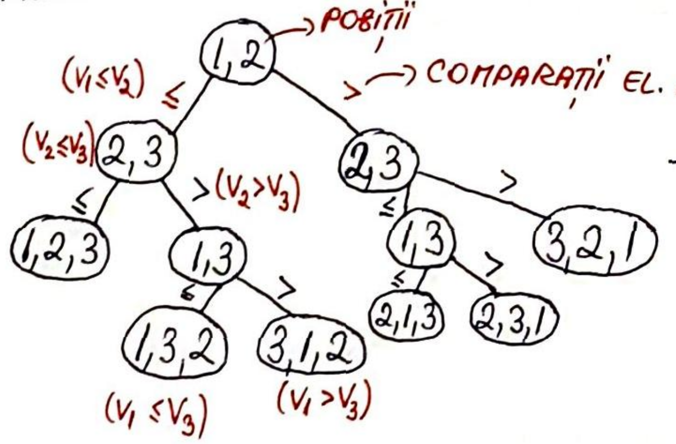
802 _ _2 _24 _45 _66 170 _75 _90

2 24 45 66 75 90 170 802 ⇒ SORTAT

ORICE ALGORITM DE SORTARE BAZAT PE COMPARAȚII ÎNTRE CHEI ARE TIMP DE RULARE (ÎN CEL MAI DEFAVORABIL CAZ) =

$\boxed{\Omega(n \cdot \log n)}$

ALGORITMII DE SORTARE BAZAȚI PE COMPARAȚIA DINTRE CHEI POT FI REPREZENTAȚI CA ARBORI DE DECIZIE.



⇒ ARB. DE DECIZIE PT. 3 ELEMENTE. SUNT POSIBILE $3! = 6$ = PERMUTĂRI (= = FRUNZE)

ÎN CEL MAI DEFAVORABIL CAZ, NR. DE COMPARAȚII REALIZAT DE ARBORELE DE DECIZIE ESTE EGAL CU CEL MAI LUNG DRUM DE LA RĂDĂCINĂ, LA FRUNZE; ADICĂ ESTE EGAL CU ÎNĂLȚIMEA ARBORELUI.

ARB. BINAR CU ÎNĂLȚIMEA "h" NU ARE MAI MULT DE 2^h FRUNZE.

$n! \leq 2^h \mid \log_2$

$\log_2 n! \leq \log_2 2^h = h$

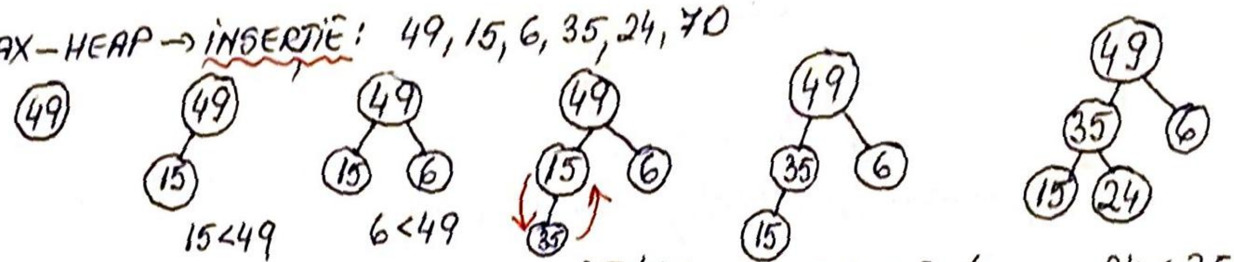
PRIN APROXIMAREA LUI STIRLING $\left(n! \geq \left(\frac{n}{e}\right)^n\right)$ ⇒ $h \geq \log\left(\frac{n}{e}\right)^n = n \cdot \log n - n \log e = \Omega(n \cdot \log n)$

= HEAP-URI =

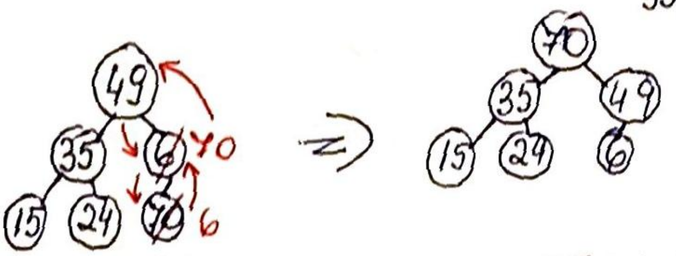
HEAP → STRUCT DATE CE PERMITE:
1) INSERȚIA ⇒ $O(\log n)$
2) RETURN MAX/MIN (= RĂDĂCINA) → $O(1)$
3) ȘTERGERE " —— " —— " ⇒ $O(\log n)$

- MAX-HEAP: NODUL MAX = RĂDĂCINĂ (TOȚI FII SUNT MAI MICI DECÂT NODUL TATĂ)
- MIN-HEAP: NODUL MIN = RĂDĂCINĂ

EX: MAX-HEAP → INSERȚIE: 49, 15, 6, 35, 24, 70



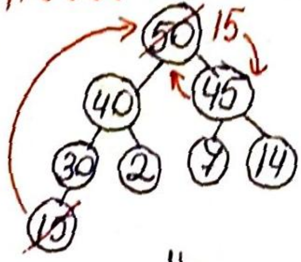
15<49 6<49 35 ≮ 15 35<49 ✓ 24 < 35



70 ≮ 6
70 ≮ 49

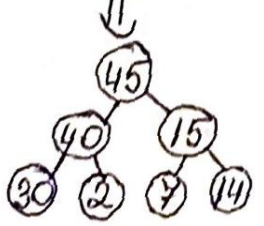
$\underbrace{70}_{0}, \underbrace{35}_{1}, \underbrace{49}_{2}, \underbrace{15}_{3}, \underbrace{24}_{4}, \underbrace{6}_{5}$

ȘTERGERE MAXIM DIN MAX-HEAP:



- ȘTERGEM RĂDĂCINA (MAXIMUL)
- PUNEM $\underbrace{\text{ULTIMA FRUNZĂ}}_{\text{ULTIMUL ELEMENT DIN VECTOR}}$ ÎN LOCUL LIBER
- REARANJĂM HEAP-UL PT. CĂ NU MAI RESPECTĂ REGULA MAX (INTERSCHIMB REPETAT CU NODURILE DIN DREAPTA)

⇓



POZIȚII VECTOR: PT. NOD "i"
- TATĂL = $[i/2]$
- FIU STÂNG = $2i$
- FIU DREPT = $2i+1$

	1	2	3	4	5	6	7
EX:	45	40	15	30	2	7	14

$i = 2 \Rightarrow$ NOD 40 ⇒
- TATĂL = $i/2 = 1 \Rightarrow 45$
- FIU ST = $2i = 4 \Rightarrow 30$
- FIU DR = $2i+1 = 5 \Rightarrow 2$

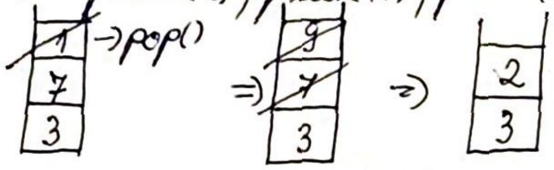
= STIVE. COZI =

1) STIVĂ

• LIFO = LAST IN, FIRST OUT. OPERAȚIILE SE EXECUTĂ LA ACELAȘI CAPĂT.

OPERAȚII: { push (NR) → INSERT
pop() → DELETE VÂRF

EX: push(3); push(7); push(1); pop(); push(9); pop(); pop(); push(2)



2) COADĂ

• FIFO = FIRST IN, FIRST OUT. OPERAȚIILE SE EXECUTĂ LA CAPETE DIFERITE.

Ex: push (1); push(7); push(3); pop(); push(9); pop(); pop(); push(2)

IN: 9 3 7 ~~1~~ :OUT

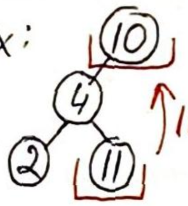
IN: 2 9 ~~3~~ ~~7~~ :OUT

IN: 2 9 :OUT

= ARBORE BINAR DE CĂUTARE =

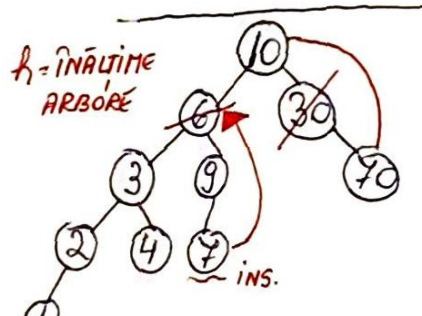
DEF: A.B.C. = UN ARBORE BINAR (UN NOD ARE MAX 2 FII) ÎN CARE FIECARE NOD ARE VALOAREA MAI MARE DECÂT TOATE NODURILE DIN SUBARBORELE STÂNG ȘI ARE " —— " —— " MICĂ " —————— " —————— " DREPT.

Ex:



$11 > 10$ ⇒ NU E A.B.C.

OPERAȚII:
1) INSERȚIE
2) CĂUTARE
3) ȘTERGERE
4) MIN/MAX
5) SUCCESOR/PREDECESOR
6) PARCURGERI



1) INSERȚIE: pt 7 $\boxed{\Rightarrow \in O(h)}$

$7 < 10$ → stânga; $7 > 6$ → dreapta; $7 < 9$ → stânga

2) CĂUTARE: pt. 2 → $\boxed{\in O(h)}$; h = ÎNĂLȚIME ARB.

$2 < 10$ → st; $2 < 6$ → st; $2 < 3$ → st; $2 = 2$ → GASIT

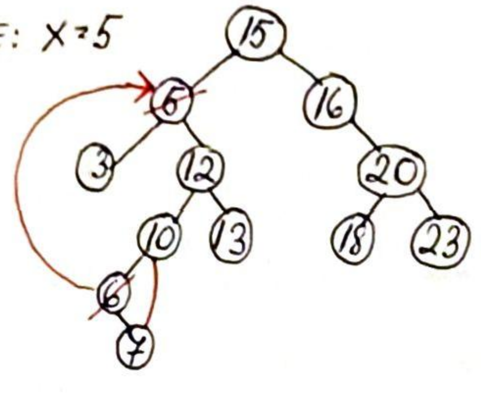
3) ȘTERGERE: a) "X" = FRUNZĂ (NU ARE FII) → TRIVIAL (EX: 1)

b) "X" ARE UN SINGUR FIU (ST/DR) (EX: 30): SE ELIMINĂ "X" ȘI SE FACE LEGĂTURA DINTRE TATĂL LUI "X" ȘI FIUL LUI.

c) "X" ARE AMBII FII (EX 6): SE ÎNLOCUIEȘTE "X" CU SUCCESORUL (">X") CARE NU ARE FIU STÂNG.

⇒ $\boxed{\in O(h)}$

ALT EX. PT. ȘTERGERE: X = 5



CĂUTĂM SUCCESOR FĂRĂ FIU STÂNG, DAR AVEM GRIJĂ SĂ PARCURGEM ARBORELE DE LA STÂNGA LA DREAPTA:

- 12 > 5, DAR ARE FIU ST;
- 10 , " —"— "
- 6 , NU ARE FIU ST.

4) MIN → COBORÂM PE STÂNGA PÂNĂ LA NULL (EX: 3)
MAX → " — " — " DREAPTA " — " — " (EX: 23)

$\Rightarrow \boxed{\in O(h)}$

5) SUCCESOR: CEL MAI MIC NOD MAI MARE DECÂT "X" (EX: SUCCESOR 10 = 12)

a) "X" ARE FIU DR ⇒ SUCCESOR = MIN DIN SUBARBORE DR.

b) "X" NU " — " — " ⇒ " — " — " = CEL MAI JOS "y" A.Î. "y" ȘI FIU ST. "y" SUNT STRĂMOȘI PT. "X".

PREDECESOR: CEL MAI MARE NOD MAI MIC DECÂT "X" (EX: PREDECESOR 10 = 7)

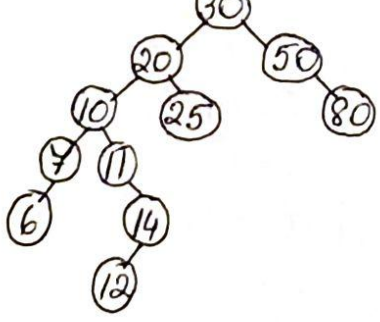
$\rightarrow \boxed{\in O(h)}$

6) PARCURGERI:

RSD (RĂDĂCINĂ-STÂNGA-DREAPTA): 30, 20, 10, 7, 6, 11, 14, 12, 25, 50, 80

*SRD: 6, 7, 10, 11, 12, 14, 20, 25, 30, 50, 80 → OBȚINEM VECTORUL ORDONAT

SDR: 6, 7, 12, 14, 11, 10, 25, 20, 80, 50, 30



ÎNĂLȚIME ARBORE COMPLET $\in O(\log n)$:

n = NR. NODURI

l_k = " — " — " PE NIVEL "K" $\begin{cases} l_k = 2 \cdot l_{k-1} \\ l_0 = 1 \text{ (RĂDĂCINA)} \end{cases}$

$2^0 + 2^1 + \dots + 2^k = n = 2^{k+1} - 1 \Rightarrow 2^{k+1} = n+1 \Rightarrow \log_2 2^{k+1} = \log_2 (n+1) \Rightarrow$

$\Rightarrow k+1 = \log_2 (n+1) \Rightarrow k = \log_2 (n+1) - 1 \Rightarrow k \in O(\log n)$

A.B.C.. DEM. CĂ DACĂ UN NOD "X" ARE FIU DREPT, ATUNCI SUCCESORUL SĂU NU ARE FIU STÂNG. #

PRIN ABSURD, (∃) y a.î. y = FIU STÂNG AL SUCC(X).

y = FIU STÂNG SUCC(X) ⇒ y < SUCC(X)

y ȘI SUCC(X) SUNT ÎN SUBARBORE DREPT X ⇒ y, SUCC(X) > X

⇒ ↯ (y == SUCC(X))

2) ARBORE BINAR PLIN = (∀) NOD (CU EXCEPȚIA FRUNZELOR) ARE EXACT 2 FII.

DEM. CĂ (∀) ARBORE NEPLIN, NU POATE CORESPUNDE UNUI COD OPTIM. #

ARB. OPTIM = ARE COSTUL MINIM.

$$\text{COST(ARBORE)} = \sum_{i \in \text{FRUNZĂ}} \text{FRECVENȚĂ}(i) \cdot \text{ADÂNCIME}(i)$$

$\boxed{2^h - 1 = \text{NR. NODURI ARB. BINAR PLIN}}$

PRESUPUN "T" = ARB. NEPLIN. PRIN ABSURD, E OPTIM..

T = NEPLIN ⇒ (∃) NODUL $v \in T$ CARE ARE UN SINGUR FIU. DACĂ ELIMIN v DIN T, ADÂNCIMEA TUTUROR NODURILOR DIN SUBARBORELE LUI "v" VA SCĂDEA ⇒

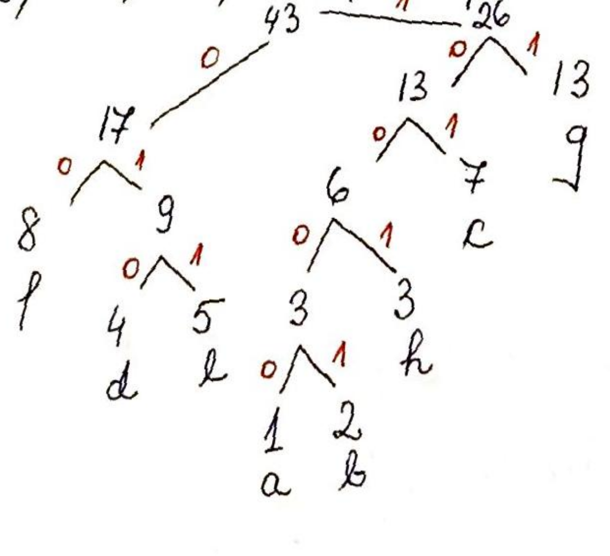
⇒ cost(T − {v}) < cost T ↯ (PT. CĂ L-AM PRESUPUS PE "T" OPTIM)

= ARBORI HUFFMAN =

Ex: $f=8$; $a=1$; $g=13$; $h=3$; $c=7$; $b=2$; $d=4$; $e=5$

LE ORDONĂM:

$a=1$; $b=2$; $h=3$; $d=4$; $e=5$; $c=7$; $f=8$; $g=13$



$a = 10000$
$b = 10001$
$c = 101$
$d = 010$
$e = 011$
$f = 00$
$g = 11$
$h = 1001$

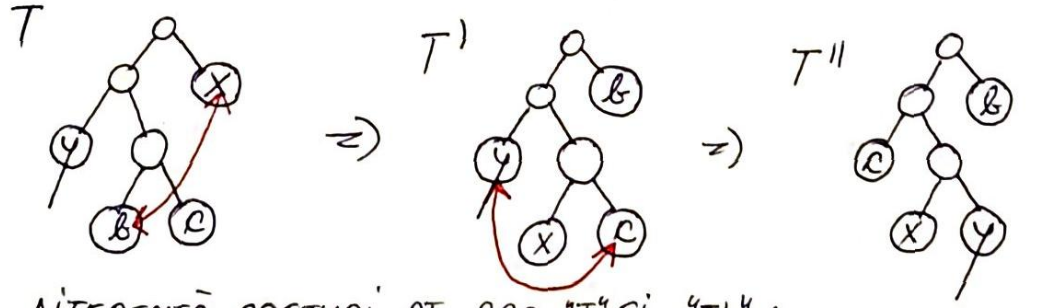
DEMONSTRAȚIE CORECTITUDINE ALGORITM HUFFMAN – METODA GREEDY

PT. A DEM. CĂ ALG. HUFFMAN E CORECT, VOM ARĂTA CĂ PROBLEMA DETERMINĂRII UNEI CODIFICĂRI PREFIX OPTIME IMPLICĂ ALEGERI GREEDY ȘI ARE O STRUCTURĂ OPTIMĂ. URMĂTOAREA LEMĂ SE REFERĂ LA PROP. ALEGERII GREEDY.

LEMA 1: FIE "C" UN ALFABET ÎN CARE FIECARE CARACTER $c \in C$ ARE O FRECVENȚĂ $f[c]$. FIE "x" ≠ "y" $\in C$, AVÂND CELE MAI MICI FRECVENȚE. ATUNCI (∃) O CODIFICARE PREFIX OPTIMĂ PT "C" ÎN CARE CUVINTELE DE COD PT. "x" ȘI "y" AU ACEEAȘI LUNGIME ȘI DIFERĂ DOAR PRIN ULTIMUL BIT.

LUĂM "T" = ARB., REPREZENTÂND O CODIFICARE PREFIX OPTIMĂ ȘI-L MODIFICĂM, REALIZÂND O CODIFICARE PREFIX OPTIMĂ. ÎN NOUL ARB., "x" ȘI "y" VOR APĂREA CA FRUNZE CU ACELAȘI TATĂ ȘI SE VOR AFLA PE NIVELUL MAX DIN ARB. ⇒ ⇒ "x" ȘI "y" VOR AVEA UN BIT DIFERIT (ULTIMUL BIT). PRESUPUN $f[x] \le f[y]$. (1)
FIE "b" ȘI "c" = FRUNZE FRAȚI, SITUATE PE NIVEL MAX ARB., CU FRECVENȚE ARBITRARE. PRESUPUN $f[b] \le f[c]$. (2)

$\overset{(1)+(2)}{\Rightarrow} f[x] \le f[b]$ ȘI $f[y] \le f[c]$



DIFERENȚĂ COSTURI PT. ARB. "T" ȘI "T'":

$$B(T)-B(T') = \sum_{c\in C} \underbrace{f[c]}_{\text{FRECVENȚĂ}} \cdot d_T(c) - \sum_{c \in C} f[c]\cdot \underbrace{d_{T'}(c)}_{\text{ADÂNCIME}} =$$

$$= f[x]\cdot d_T(x) + f[b]\cdot d_T(b) - f[x]\, d_{T'}(x) - f[b]\cdot d_{T'}(b) =$$

$$= f[x]\cdot d_T(x) + f[b]\cdot d_T(b) - f[x]\cdot d_T(b) - f[b]\cdot d_T(x) =$$

$$= (f[b]-f[x])(d_T(b)-d_T(x)) \ge 0$$ ⇒ INTERSCHIMBAREA NU MĂREȘTE COSTUL

ANALOG, $B(T)-B(T'') \ge 0 \Rightarrow B(T'') = B(T) \Rightarrow T''$ = ARB. OPTIM ÎN CARE "x" ȘI "y" SUNT FRAȚI ȘI SE AFLĂ PE NIVEL MAX (CONFORM LEMEI 1).

DIN LEMA 1 ⇒ PROCESUL DE CONSTRUCȚIE A UNUI ARB. OPTIM PRIN FUZIONĂRI POATE SĂ ÎNCEAPĂ CU O ALEGERE GREEDY A FUZIONĂRII CELOR 2 CARACTERE CU CEA MAI REDUSĂ FRECVENȚĂ.

URMĂTOAREA LEMĂ ARATĂ CĂ PB. CONSTRUIRII UNEI CODIFICĂRI PREFIX OPTIME ARE PROP. SUBSTRUCTURII OPTIME.

LEMA 2: "T" = ARB. BINAR COMPLET, REPREZENTÂND O CODIFICARE PREFIX OPTIMĂ PESTE UN ALFABET C. "X" și "Y" ∈ C, NODURI TERMINALE FRAȚI ÎN T și "Z" = TATĂL LOR. PRESUPUN $f[z] = f[x] + f[y]$. ATUNCI $T' = T - \{x, y\}$, REPREZENTÂND O CODIFICARE PREFIX OPTIMĂ PT. ALFABETUL $C' = C - \{x, y\} \cup \{z\}$.

PT. FIECARE $c \in C - \{x, y\}$, AVEM: $\begin{cases} d_T(c) = d_{T'}(c) \\ f[c] \cdot d_T(c) = f[c] \cdot d_{T'}(c) \end{cases}$

PT. $d_T(x) = d_T(y) = d_{T'}(z) + 1. \Rightarrow f[x] \cdot d_T(x) + f[y] \cdot d_T(y) = (\underbrace{f[x] + f[y]}_{f[z]})(d_{T'}(z) + 1) =$
$= f[z] \cdot d_{T'}(z) + (f[x] + f[y]) \Rightarrow B(T) = B(T') + f[x] + f[y]$

DACĂ T' NU E O CODIFICARE OPTIMĂ PT. C', ATUNCI (∃) UN ARB. T'' ALE CĂRUI FRUNZE ∈ C' A.Î. $B(T'') < B(T')$. CUM $z \in C' \Rightarrow z$ = FRUNZĂ ÎN T''. DACĂ ADĂUGĂM "X" și "Y" ÎN T'' CA FII A LUI "Z", ATUNCI VOM OBȚINE O CODIFICARE PREFIX PT. "C", CU COSTUL $B(T'') + f[x] + f[y] < B[T]$, CEEA CE INTRĂ ÎN CONTRADICȚIE CU FAPTUL CĂ T = OPTIM $\Rightarrow T'$ = OPTIM PT. C'.

= GĂSIREA MEDIANEI ÎN $O(n)$ =
(ÎMBUNĂTĂȚIRE QUICKSORT)

DATE GENERALE:

MEDIANA = AL $[\frac{n}{2}]$-LEA ELEMENT DINTR-UN VECTOR (CU "n" ELEMENTE) DEJA SORTAT. SOLUȚII PT. A GĂSI MEDIANA:

- $O(k \cdot n) \to$ TRIVIAL ($k = \frac{n}{2}$)
- $O(n \cdot \log n) \to$ SORTĂM CU MERGE/HEAP/QUICK-SORT ȘI ALEGEM AL $[\frac{n}{2}]$ EL..

CUM ALEGEM ÎN $O(n)$?

VARIANTA 1: ALG. PROBABILIST $\Rightarrow$ TIMP MEDIU = $O(n)$

```
selectie_aleator(A[], p, n, k)
{ if(p==n) returne A[p]; -> MEDIANA
  q = partitie_aleatoare(A[], p, n); //DE LA QSORT
  k1 = q-p+1; // EL. "≤" PIVOT
  if(k==k1) returne A[k];
  if(k1>k) returne selectie_aleator(A, q+1, n, k-k1);
  returne selectie_aleator(A, p, q-1, k);
}
```

$$T(n) \leq \frac{1}{n}\left(T(\max(1, n-1)) + \sum_{k=1}^{n-1} T(\max(k, n-k))\right) + O(n) \leq$$

$$\leq \frac{1}{n}\left(T(n-1) + 2\sum_{k=[n/2]}^{n-1} T(k)\right) + O(n) = \frac{2}{n}\sum_{k=[n/2]}^{n-1} T(k) + O(n)$$

$\max(1, n-1) = n-1$

$$\max(k, n-k) = \begin{cases} k; \text{ dacă } k \geq [n/2] \\ n-k; \text{ dacă } k < [n/2] \end{cases}$$

DACĂ $\begin{cases} n = \text{IMPAR} \Rightarrow T([n/2]), T([n/2]+1) \ldots T(n-1) \text{ APAR DE 2 ORI ÎN SUMĂ} \\ n = \text{PAR} \Rightarrow T([n/2]) \text{, " —— " —————— "} \end{cases}$

$T([n/2])$ APARE O SINGURĂ DATĂ

ÎN CAZUL CEL MAI DEFAVORABIL, $T(n-1) = O(n^2) \Rightarrow \frac{1}{n}T(n-1) \in O(n)$

TB. SĂ DEM. $T(n) \in O(n) \Leftrightarrow T(n) \leq c \cdot n$

$$T(n) \leq \frac{2}{n}\sum_{k=[n/2]}^{n-1} c \cdot k + O(n) = \frac{2c}{n}\left(\sum_{k=1}^{n-1} k - \sum_{k=1}^{[n/2]-1} k\right) + O(n) =$$

$$= \frac{2c}{n}\left(\frac{1}{2}(n-1)n - \frac{1}{2}\left(\left[\frac{n}{2}\right]-1\right)\left[\frac{n}{2}\right]\right) + O(n) \leq c(n-1) - \frac{c}{n}\left(\frac{n}{2}-1\right)\left(\frac{n}{2}\right) + O(n) =$$

$$= c\left(\frac{3}{4}n - \frac{1}{2}\right) + O(n) \leq c \cdot n$$

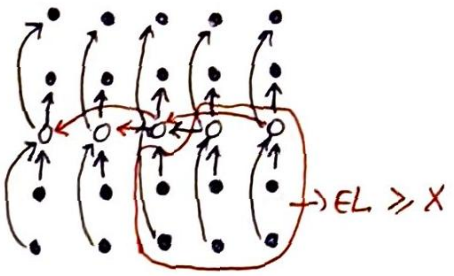
VARIANTA 2: ALG. DETERMINIST $\Rightarrow$ WORST TIME: $O(n)$

1) SE ÎMPART ELEMENTELE ÎN GRUPE DE 5. $\to O(n)$

2) SE DETERMINĂ MEDIANA FIECĂRUI GRUP. $\to O(n)$

3) APELEZ RECURSIV ALGORITMUL, PT. A DETERMINA MEDIANA MEDIANELOR. CONSIDER "X" = MEDIANA MEDIANELOR.

4) FOLOSESC "X" CA PIVOT ȘI ÎMPART FOLOSIND ALG. PROBABILIST.



AVEM $\frac{n}{5}$ GRUPE.

$3 \cdot \frac{1}{2} \cdot \frac{n}{5} = \frac{3n}{10}$ ELEMENTE "$>$" X

ASTFEL, ÎN CEL MAI DEFAVORABIL CAZ, VOI APELA RECURSIV ALG. PROBABILIST PT.

$n - \frac{3n}{10} = \frac{7n}{10}$ ELEMENTE.

$T(n) = T(n/5) + T(7n/10) + O(n) \in O(n)$

PRESUPUN $\begin{cases} T(\frac{n}{5}) \leq c \cdot \frac{n}{5} \\ T(\frac{7n}{10}) \leq c \cdot \frac{7n}{10} \end{cases}$

TB. SĂ DEM: $T(n) \leq c \cdot n$

$T(n) = T(n/5) + T(7n/10) + n \leq c \cdot \frac{2n}{5} + c \cdot \frac{7n}{10} + n = c \cdot \frac{9n}{10} + n = n(1 + \frac{9c}{10}) \leq$

$\leq c \cdot n$, (A) pt (∀) $c \geq 10$

---

a) PT. GRUPE DE 3:

$\frac{n}{3}$ GRUPE $\Rightarrow \cancel{2} \cdot \frac{1}{\cancel{2}} \cdot \frac{n}{3} = \frac{n}{3}$ EL. "$\geq$" X $\Rightarrow n - \frac{n}{3} = \frac{2n}{3}$

$T(n) = T(n/3) + T(2n/3) + O(n) \in O(n \cdot \log n)$

b) PT. GRUPE DE 7:

$\frac{n}{7}$ GRUPE $\Rightarrow \cancel{4}^{2} \cdot \frac{1}{\cancel{2}} \cdot \frac{n}{7} = \frac{2n}{7} \Rightarrow n - \frac{2n}{7} = \frac{5n}{7}$

$T(n) = T(n/7) + T(5n/7) + O(n) \in O(n)$

EXERCIȚII:

1) AVEM "K" LISTE SORTATE CU "n" ELEMENTE TOTALE. SĂ SE INTERCLASEZE CÂT MAI EFICIENT.

$O(n \cdot \log n)$: CONCATENEZ LISTELE ȘI SORTEZ CU MERGESORT

$O(n \cdot K)$: LUĂM MIN DE PE FIECARE LISTĂ ȘI INDICE PT. FIECARE LISTĂ

$O(n \cdot \log K)$: FOLOSIM UN MIN-HEAP (PUNEM ELEMENTELE ÎN HEAP ȘI EXTRAGEM VALORILE CELE MAI MICI

2) "n" NR. VREM CELE MAI APROPIATE "K" VALORI DE MEDIANĂ.

$O(n \cdot \log n)$: QSORT + GĂSIRE ELEMENTE

$O(n)$: GĂSIM MEDIANA "X" ÎN $O(n)$

CONSTRUIM UN VECTOR $v_i = |a_i - X|$

GĂSIM A "K"-A VALOARE DIN VECTORUL "v_i".

APLICĂM FUNCȚIA "PARTITION" DUPĂ ACEASTĂ VALOARE

EX: 7 5 24 18 11 3 41 64 2 $K=3$

$|7-11|=4$ v: 4 6 13 7 0 8 30 53 9

$|5-11|=6$ ② ③ ①

⋮

3) NR. INVERSIUNI PERMUTARE = ?

APLICĂM UN MERGESORT MODIFICAT.

$INV(s, d)$

$INV(s, \frac{d}{2}) \to$ NR INVERSIUNI 1/2 JUMĂTATE

$INV(\frac{d}{2}+1, d) \to$ " —"— " A 2-A "—"—"

$COMBINA\_SOL(s, \frac{d}{2}, \frac{d}{2}+1, d) \to$ CALCULEAZĂ NR. INV. DIN CELE 2 JUMĂTĂȚI, VERIFICÂND ÎN TIMPUL INTERCLASĂRII CÂTE NR. DIN A 2-A JUMĂTATE SUNT MAI MICI DECÂT CELE DIN PRIMA.

4) SE DĂ UN SET DE NR.. VREM MIN ȘI AL II-lea MIN., CU NR. MINIM DE COMPARAȚII.

5 7 2 9 8 11 0 39

5 2 8 0

$\boxed{2}$ $\boxed{0}$

LUĂM NR ÎN PERECHI ȘI FACEM MINIMUL PE FIECARE PERECHE.

APOI, RECURSIV, APLIC ACEEAȘI METODĂ PÂNĂ RĂMÂN CU 2 NR. $\Rightarrow (n-1)$ COMPARAȚII

$\frac{n}{2} + \frac{n}{4} + \ldots + 1 = (n-1) \Rightarrow O(\log n)$

5) SE DAU "n" GĂLEȚI ROȘII ȘI "n" GĂLEȚI GALBENE: $\begin{cases} r_1 \dots r_n \in [x; y] \\ g_1 \dots g_n \in [x; y] \end{cases}$

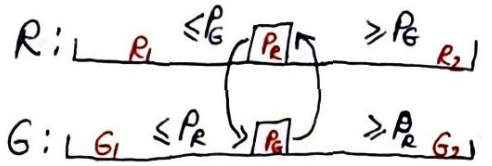
$(\forall)\ i \in \overline{1,n}$ VREM SĂ SE GĂSEASCĂ j a.î. $r_i = g_j$

FĂRĂ SĂ COMPARĂM GĂLEȚILE CU ACEIAȘI CULOARE.

$O(n^2)$: 2 for-uri (PT. FIECARE GALBEN, CAUT ROȘU)

SAU

MAI EFICIENT: FOLOSESC PIVOTUL CU ACEEAȘI VALOARE DIN SETUL CELĂLALT ȘI SORTEZ. ($P_R \to$ SORTEZ GALBENE; $P_G \to$ SORTEZ ROȘII)



6) SE DAU "n" NR $\in [0; n^3]$. SĂ SE SORTEZE NR ÎN $O(n)$.

TRANSFORM NR ÎN BAZA "n".

NR $\in [0; n^3] \Rightarrow$ ÎN BAZA "n" VOR AVEA MAX 3 CIFRE DIN INTERVALUL $\overline{0, n-1}$

FOLOSESC RADIXSORT (COUNTINGSORT DREPT SUBRUTINĂ) $\Rightarrow O(n)$

$$\hookrightarrow b \cdot (n+k) = b \cdot (n+n) = 2\underset{3}{\underset{\shortparallel}{b}}n = 6n \in O(n)$$

7) UN ȘIR DE NR. CU "n" CIFRE ÎN TOTAL.

VREM SĂ SORTĂM ÎN $O(n)$.

$n = 17$

12345		00051
00051	RADIX SORT $\to$	00213
00999		00999
01234		01234
00213		12345

DEMONSTRĂM CĂ $\in O(n)$.

$\#i$ = NR. DE VALORI DIN INPUT CARE AU "$\ge$" CU "i" CIFRE.

EX: $i = 4 \Rightarrow \#i = 2$ (1234, 12345)

LA PAS 1: $\#1 +$ 10 $\to$ DIN COUNTINGSORT

⋮

LA PAS n: $\#n + 10$

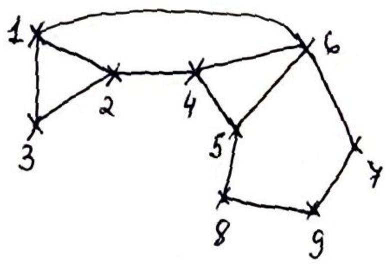
$\oplus$

$\#1 + \#2 + \dots + \#n + 10 \cdot n = n + 10n = 11n \in O(n)$

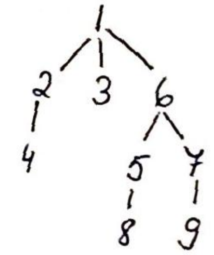
= EXERCIȚII GRAFURI =

1) METODE DE PARCURGERE:

- DFS (DEPTH FIRST SEARCH) → ÎN ADÂNCIME: 123456798
  ↓
  (STIVĂ)



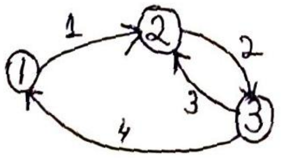
- BFS (BREADTH F. S.) → ÎN LĂTIME: 123645789 ⟺
  ↓
  (COADĂ)



2) GRAF ORIENTAT. DEFINIM MATRICEA "B" DE DIMENSIUNEA NR MUCHII × NR VF.

$$B_{ij} = \begin{cases} -1, & \text{MUCHIA } j \text{ IESE DIN NOD } i \\ 1, & \text{—"— INTRĂ ÎN —"—} \\ 0, & \text{ALTFEL} \end{cases}$$

CALCULAȚI $B \cdot B^T$ PE UN EXEMPLU:



B	1	2	3	4
1	-1	0	0	1
2	1	-1	1	0
3	0	1	-1	-1

$$\begin{pmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 1 & 0 \\ 0 & 1 & -1 & -1 \end{pmatrix} \cdot \begin{pmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \\ 1 & 0 & -1 \end{pmatrix} = \begin{pmatrix} \boxed{2} & -1 & 1 \\ -1 & \boxed{3} & -2 \\ -1 & -2 & \boxed{3} \end{pmatrix}$$

→ DIAGONALĂ PRINCIPALĂ = GRAD INTERN + EXTERN FIECARE NOD

3) DEM. CĂ ÎNTR-UN GRAF NEORIENTAT EXISTĂ 2 NODURI CU ACELAȘI GRAD.

CAZ 1: (∃) 1 NOD IZOLAT

n-1 NODURI → n-2 GRADE ⇒ EVIDENT

CAZ 2: (∄) NOD IZOLAT

n NODURI → n-1 GRADE ⇒ —"—

4) SE DAU "n" MATRICI $A_1, A_2, \dots, A_n$. SĂ SE GĂSEASCĂ O PARANTEZĂ A.Î. LA ÎNMULȚIREA MATRICELOR NR. DE PAȘI SĂ FIE CÂT MAI MIC.

$$\left.\begin{array}{l} A_{n \times m} \\ B_{m \times k} \end{array}\right\} \Rightarrow C_{n \times k}$$

$A(B\ C\ D\ E\ F)$

$(A\ B)(C\ D\ E\ F)$

$(A\ B\ C)(D\ E\ F)$

$i \quad k+1 \quad j+1$

$OPTIM(i,j)$ = COST OPTIM ÎNMULȚIRI $A_i A_{i+1} \dots A_j$

$$OPTIM(i,j) = \min_{i \le k < j}\left(OPTIM(i,k) + OPTIM(k+1,j) + \underbrace{a_i a_{k+1} a_{j+1}}_{\text{COST DE ÎNMULȚIRE CELE 2 MATRICI REZULTATE DUPĂ PARANTEZĂ}}\right)$$

Ex: $OPTIM(1,3) = \min\big(OPTIM(1,1) + OPTIM(2,3) + a_1 a_2 a_4,$
$OPTIM(1,2) + OPTIM(3,3) + a_1 a_3 a_4\big)$

= SIMULARE EXAMEN ASC =

1) a) $33n^8 + 44\sqrt{n^{15}} + 55\sqrt[3]{n} + 66 = \Theta(n^8) = \Omega(n^7) = O(n^9)$

b) $77\ln(n^{88}) + 99\underbrace{\sin\sqrt{n+n^{100}}}_{\in[-1,1]} + \sqrt[3]{n} = 77\cdot 88\cdot \ln n + \sqrt[3]{n} = \Theta(\sqrt[3]{n}) = \Omega(\ln n) = O(n^2)$

2) (T. MASTER)

a) $T(n) = 3T\left(\frac{n}{2}\right) + n\log_{\sqrt{2}} n$

$a = 3$

$b = 2$

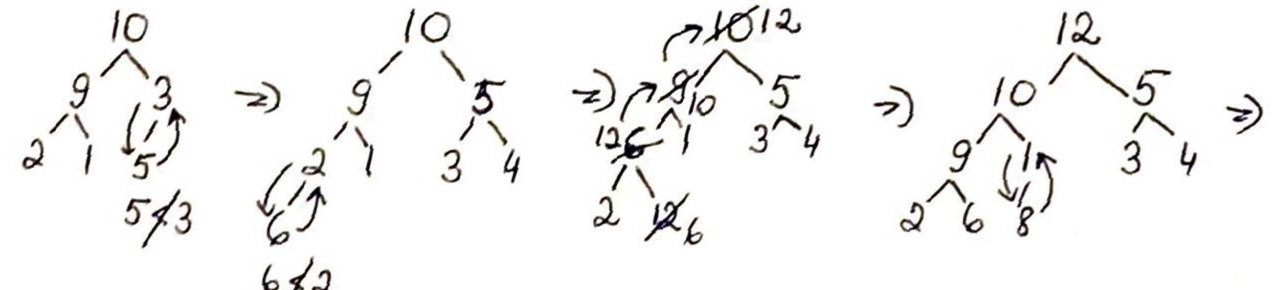
$f(n) = n\cdot\log_{\sqrt{2}} n \in O(n^{\log_2 3}) \xrightarrow{CAZ\ 1} T(n) = \Theta(n^{\log_2 3})$

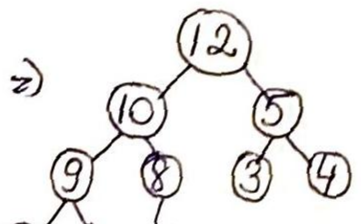
b) $T(n) = 4T\left(\frac{n}{2}\right) + n^2$

$\left.\begin{array}{l} a = 4 \\ b = 2 \end{array}\right\} \Rightarrow n^{\log_b a} = n^2$

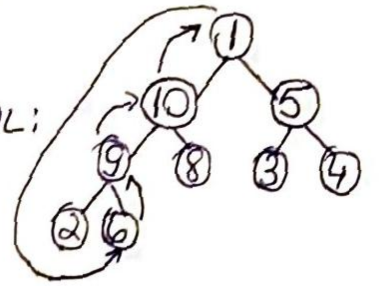
$f(n) = n^2 \in \Theta(n^2) \xrightarrow{CAZ\ 2} T(n) = \Theta(n^2\cdot\log_2 n)$

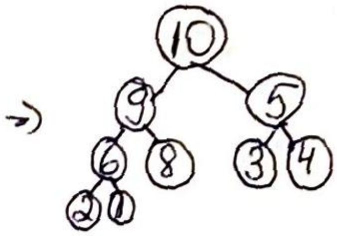
3) MAX-HEAP PT: 10, 9, 3, 2, 1, 5, 4, 6, 12, 8





ȘTERGEȚI MAXIMUL:





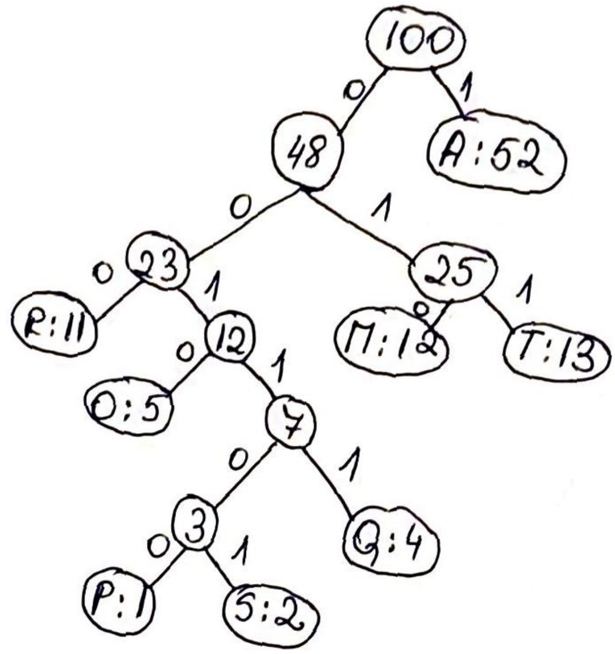
4) (HUFFMAN)

A	P	S	O	Q	M	T	R
52	1	2	5	4	12	13	11

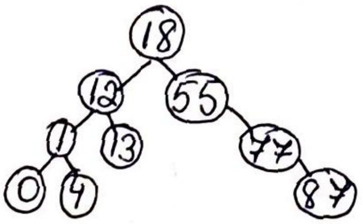
→

P	S	Q	O	R	M	T	A
1	2	4	5	11	12	13	52

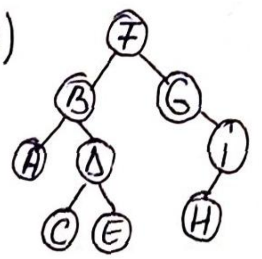
P: 001100
S: 001101
Q: 00111
O: 0010
R: 000
M: 010
T: 011
A: 1



5) (ARB. BINAR CĂUTARE) 18, 12, 55, 1, 77, 0, 87, 4, 13

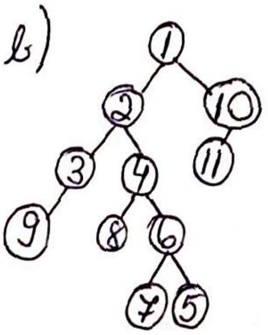


6) a)



SRD: ABCDEFGHI
SDR: ACEDBHIGF

b)



SRD: 9,3,2,8,4,7,6,5,1,11,10
SDR: 9, 3, 8, 7, 5, 6, 4, 2, 11, 10, 1