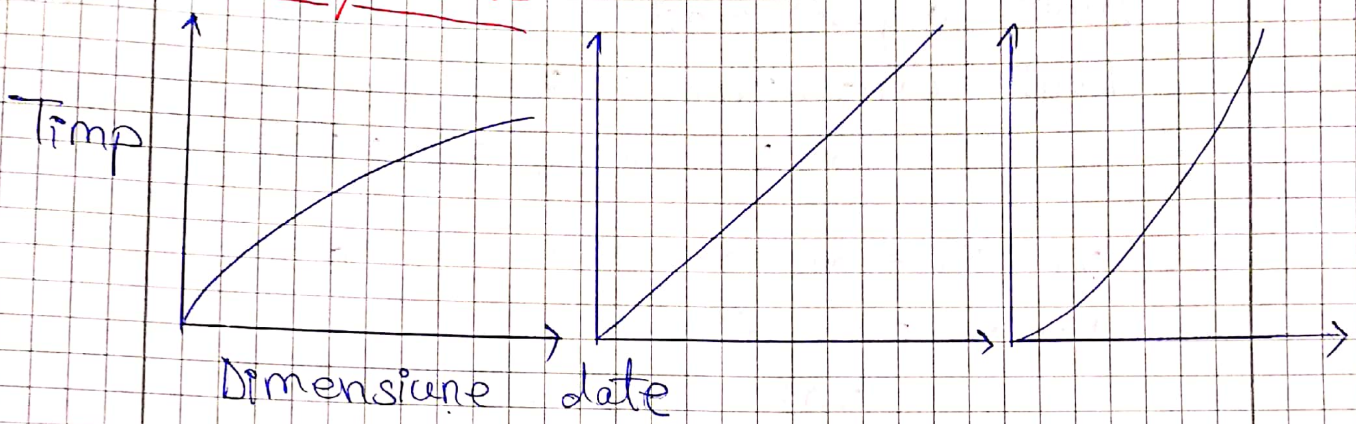
ASD

① Complexitate



Clase:

- Ω – "best case"
- O – "worst case"
- Θ – "average"

Definiții:

$f(n) = \Omega(g(n)) \iff \exists c > 0, \exists n_0 \in \mathbb{N}$ a.î.

$$f(n) \geq c \cdot g(n), (\forall) n \geq n_0$$

$f(n) = O(g(n)) \iff \exists c > 0, \exists n_0 \in \mathbb{N}$ a.î.

$$f(n) <= c \cdot g(n), (\forall) n >= n_0$$

$f(n) = \Theta(g(n)) \iff \exists c_1 > 0, \exists c_2 > 0, \exists n_0 \in \mathbb{N}$ a.î.

$$c_1 \cdot g(n) <= f(n) \leq c_2 \cdot g(n), (\forall) n >= n_0$$

Ex: $f(n) = 2n + 5$

$\rightarrow f(n) = \Omega(1)$ $(c = 1, n_0 = 3 \Rightarrow f(n) \geq 1, (\forall) n >= n_0)$

Dar vrem cea mai mare funcție

Deci $g(n) = n$ $(c = 1, n_0 = 0 \Rightarrow f(n) \geq n, (\forall) n >= n_0)$

Scanned by CamScanner

$\rightarrow f(n) = O(n^2)$ ($c = 1$, $n_0 = 4$ a.î. $f(n) \leq n^2, (\forall) n >= n_0$)

Dar merge și $g(n) = n$ ($c = 5$, $n_0 = 2$ a.î. $f(n) \leq 5n, (\forall) n >= n_0$)

Pentru a găsi constanta:

$f(n) <= c \cdot n \Rightarrow 2n+5 <= cn \Rightarrow$

$\Rightarrow c >= 2 + \frac{5}{n} \Rightarrow c >= 3$

$2n+5 <= 3n \Rightarrow 5 <= n \Rightarrow n_0 = 5$

pentru $c = 3$ și $n_0 = 5$,

$2n+5 <= 3n, (\forall) n >= 5$

V: $n = 5$, $2 \cdot 5 + 5 <= 3 \cdot 5$ - ok!

$n = 6$, $2 \cdot 6 + 5 <= 3 \cdot 6$ - ok!

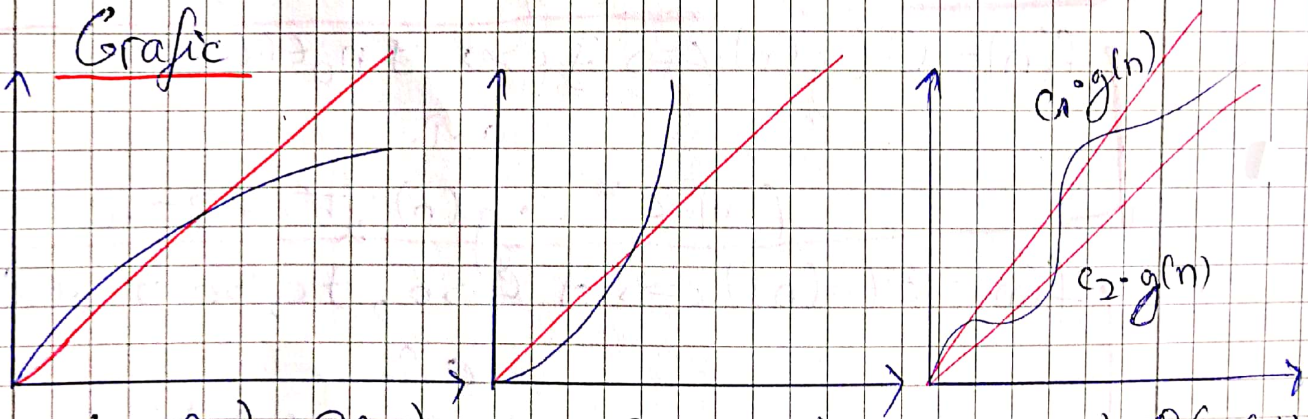
$\rightarrow f(n) = \Theta(n)$ ($c_1 = 1$, $c_2 = 3$, $n_0 = 5$ a.î.

$g(n) <= f(n) <= 3 \cdot g(n), (\forall) n >= 5$)

Nu putem găsi mereu,

dar dacă $\left.\begin{array}{l} f(n) = \Omega(g(n)) \\ f(n) = O(g(n)) \end{array}\right| \Rightarrow f(n) = \Theta(g(n))$

Grafic



$\log(n) = O(n)$ $\quad$ $2^n = \Omega(n)$ $\quad$ $f(n) = \Theta(g(n))$

Proprietăți

- $\rightarrow$ tranzitivitate
- $\rightarrow$ reflexivitate
- $\rightarrow$ simetrie (Θ)
- $\rightarrow$ antisimetrie ($O \leftrightarrow \Omega$)

Exemple de complexități

- → operații aritmetice simple, căutare folosind tabele de dispersie (hash-tables) — $O(1)$
- → căutare binară — $O(\ln(n))$
- → căutare secvențială, suma unui șir — $O(n)$
- → sortări rapide (interclasare, quick-sort) — $O(n\ln(n))$
- → sortări „naive” (bubble-sort, inserție, ...), parcurgeri de matrice pătratice de ordin n — $O(n^2)$
- → înmulțirea clasică a matricelor — $O(n^3)$
- → generare aranjamente — 2^n

Utilizarea limitelor

→ $\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f(n) = o(g(n))$

$(f(n) = O(g(n))$ și $f(n) \neq \Theta(n))$

→ $\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} \in \mathbb{R}_+ \Rightarrow f(n) = \Theta(g(n))$

→ $\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = +\infty \Rightarrow f(n) = \omega(g(n))$ $(f(n) = \Omega(g(n))$ și $f(n) \neq \Theta(n))$

Proprietăți importante

→ dacă $f(n) = \sum_{i=0}^{m} a_i n^i \Rightarrow f(n) = \Theta(n^m)$

→ $a, b \in \mathbb{R}, b > 0 \Rightarrow (n+a)^b = \Theta(n^b)$

→ $a \in \mathbb{R}, a > 1, f(n) = \Theta(g(n)) \Rightarrow a^{f(n)} = \Theta(a^{g(n)})$

→ $\sum_{k=1}^{n} \Theta(f(k)) = \Theta\left(\sum_{k=1}^{n} f(k)\right)$

→ $\sum_{k=1}^{n} k = \frac{n(n+1)}{2} = \Theta(n^2)$ serie aritmetică

→ $\sum_{k=1}^{n} x^k = \frac{x^{n+1}-1}{x-1} = \Theta(x^n), x > 1$ serie geometrică

→ $\sum_{k=1}^{n} \frac{1}{k} = \ln(n) + \Theta(1)$ serie armonică

→ $\Theta(f) = \Theta(g) \Rightarrow \Theta(\ln(f)) = \Theta(\ln(g))$.

Metode pentru funcții recursive

1. Substituție (ca la inducție)

Ex: $T(n) = \begin{cases} T(n/2) + 1, & n > 1 \\ 1, & n = 1 \end{cases}$ corespunzătoare unei căutări binare

Presupunem $O(\log n)$

Pp. $T(n/2) \le c \log(n/2)$, $c > 0$ | +1

$T(n/2) + 1 \le c \log(n/2) + 1$
Dar $T(n) = T(n/2) + 1$ $\Bigg| \Rightarrow$

$\Rightarrow T(n) = T(n/2) + 1 \le c \log(n/2) + 1 =$

$= c\left(\log(n) + \log\frac{1}{2}\right) + 1 = c \log n + (1-c) \Rightarrow$

$\Rightarrow T(n) \le c \log n$ pentru $c \ge 1$

Pt. $n_0 = 2$, $T(2) = 2 \le c \log 2$, $(\forall) c > 2$

Deci $T(n) \le c \log n$, $(\forall) n \ge n_0 = 2$, $(\forall) c \ge 2$

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

$T(n) = O(n)$

$T(n) \le cn$

$T(n) = 2T(n/2)+1 \le 2cn/2+1 = cn+1 \not\Rightarrow T(n) \le cn$

Dar pp. $T(n) \le cn-b$, $b>0$

$T(n) \le 2\left(c\frac{n}{2}-b\right)+1 = cn-2b+1 \le cn-b$ pt. $b \ge 1$

ok!

Ex $T(n) = 2T(\sqrt{n})+\log n$

Notăm $m = \log n$

$S(m) = 2S(m/2)+m$

Sol.: $S(m) = O(m \log m) \Rightarrow T(n) = O(\log n \cdot \log(\log(n)))$

2. Iterație

Turnurile din Hanoi

Ex $T(n) = \begin{cases} T(n-1)+1+T(n-1) = 2T(n-1)+1, & n>1 \\ 1, & n=1 \end{cases}$

$T(n) = 2T(n-1)+1 = 4T(n-2)+2+1 =$

$= 8T(n-3)+4+2+1 = \dots = \sum_{i=0}^{n-1} 2^i = 2^n - 1$

Deci $T(n) = O(2^n)$

Ex $\lfloor\ \rfloor$ - parte întreagă inferioară

$\lceil\ \rceil$ - parte întreagă superioară

$T(n) = \begin{cases} T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n, & n>1 \\ 1, & n=1 \end{cases}$ sortare prin interclas

Proprietate

Dacă f netedă ($\exists\, b \ge 2$ a.î. $f(bn) = O(f(n))$), $a \ge 2$,

t nedescrescătoare a.î. $t(n) = O(f(n))$, $n = a^k$ $\Rightarrow$

$\Rightarrow t(n) = O(f)$, $(\forall)\, n$

Scanned by CamScanner

Consideram $n = 2^k$:

$T(n) = 2T(n/2) + n = 2^2 T(n/4) + 2n = \dots =$

$= 2^{\log_2 n} + \log_2(n)\cdot n = n + \log_2(n)\cdot n$

Deci $T(n) = O(n\log(n))$ pt. $n = 2^k$ $\Rightarrow$

Conform proprietate

$\Rightarrow T(n) = O(n\log(n))$, $(\forall)\, n \in \mathbb{N}$

3. Master

↳ soluție directă pentru $T(n) = aT(n/b) + f(n)$

$a \geq 1,\ b > 1$

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + O(n^d)}$$

a – numărul de subprobleme, $a > 0$

b – rata de descreștere a dimensiunii subproblemelor, $b > 1$

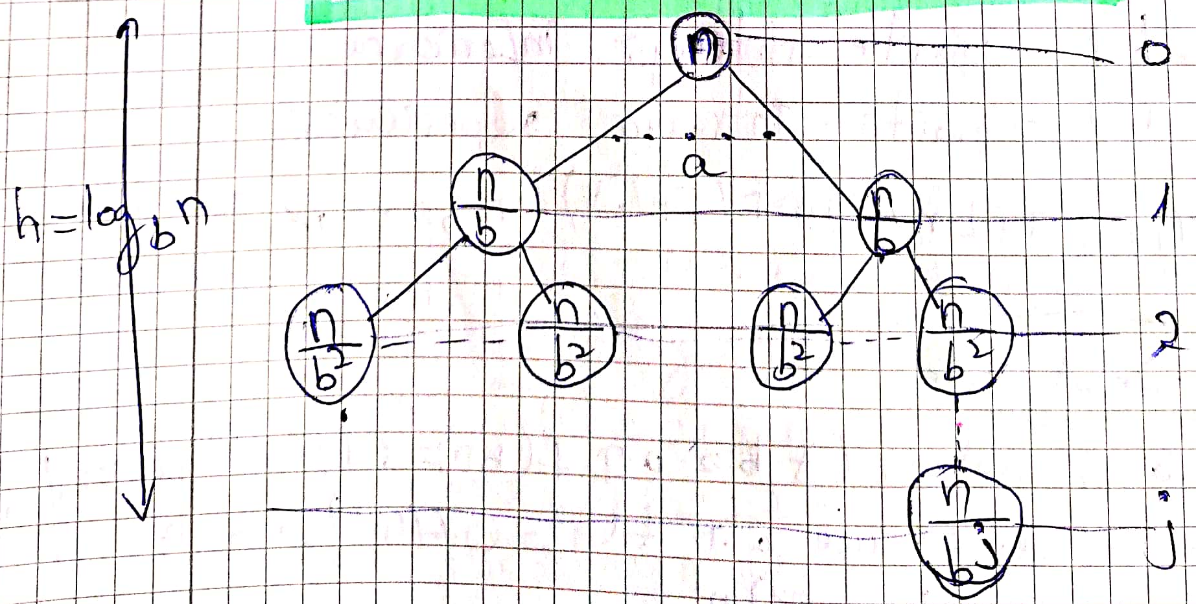
d – exponentul din Div(n) + Comb(b) $\in O(n^d)$, $d \geq 0$

$$\boxed{T(n) = \begin{cases} O(n^d \log(n)), & a = b^d \\ O(n^d), & a < b^d \\ O(n^{\log_b a}), & a > b^d \end{cases}}$$

(case 2) Θ

(caz 3) Ω

(caz 1) O



a^j = numărul de subprobleme la nivelul j

$\frac{n}{b^j}$ = dimensiunea subproblemelor de la nivelul j

$\boxed{a^{\log_b n} = n^{\log_b a} \quad \text{frunze}}$

$$T(\text{nivel } j) \le a^j \cdot O\left(\left(\frac{n}{b^j}\right)^d\right)$$
$$= a^j \cdot c \cdot \left(\frac{n}{b^j}\right)^d$$
$$= c \cdot n^d \cdot \left(\frac{a}{b^d}\right)^j$$

Avem:

(i) $T(1) \le c$

(ii) $T(n) \le a \cdot T\left(\frac{n}{b}\right) + c \cdot n^d$

$$\text{Total} = \sum_{j=0}^{\log_b n} T(\text{nivel } j) \le c \cdot n^d \cdot \sum_{j=0}^{\log_b n} \left(\frac{a}{b^d}\right)^j =$$

$$\boxed{\sum_{j=0}^{\log_b n} r^j} \qquad \boxed{r = \frac{a}{b^d}}$$

serie geometrică $\qquad$ rația

$$T(n) \le c \cdot n^d \cdot \left(\sum_{j=0}^{\log_b n} r^j\right), \quad r = \frac{a}{b^d}$$

Suma de serii geometrice

$$1 + r + r^2 + \ldots + r^k = \begin{cases} \frac{r^{k+1} - 1}{r - 1}, & \text{dacă } r \ne 1 \\ k + 1, & \text{dacă } r = 1 \end{cases}$$

(1) dacă $r = 1$, $\sum_{j=0}^{\log_b n} r^j$ = numărul de termeni = $k+1$ $(\log_b n + 1)$

(2) dacă $r < 1$, $\sum_{j=0}^{\log_b n} r^j \le \frac{1}{1-r}$

(3) dacă $r>1$, $\sum_{j=0}^{\log_b n} r^j \le r^k\left(1+\frac{1}{r-1}\right)$

Obs. $\frac{1}{1-r}$ și $1+\frac{1}{r-1}$ sunt independente de k ((2), (3))

Cazul (1) ($r=1$) $a=b^d$:

$$T(n) \le c \cdot n^d \cdot (\log_b n+1) = O(n^d \log n)$$

Cazul (2) ($r<1$) $a<b^d$

$$T(n) \le c \cdot n^d \cdot \underbrace{\left(\frac{1}{1-\frac{a}{b^d}}\right)}_{\text{constant}} = O(n^d)$$

Cazul (3) ($r>1$) $a>b^d$

$$T(n) \le c \cdot n^d \cdot \underbrace{\left(1+\frac{1}{r-1}\right)}_{\text{constant}} \cdot r^{\log_b n}$$

Din $r^{\log_b n} = \left(\frac{a}{b^d}\right)^{\log_b n} = \frac{a^{\log_b n}}{\left(b^{\log_b n}\right)^d} = \frac{a^{\log_b n}}{n^d}$

$\Downarrow$

$$T(n) \le \underbrace{O\left(a^{\log_b n}\right)}_{\text{numărul de frunze}} \quad \left(= O\left(n^{\log_b a}\right)\right)$$

Ex. Căutare binară

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

$a = 1$, $b = 2$, $d = 0$

$aT\left(\frac{n}{2}\right) + \Theta(n^d)$

$a = b^d \xrightarrow[\text{caz 1}]{} T(n) = \Theta(\log n)$

Ex. Merge Sort

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

$a = 2$, $b = 2$, $d = 1$

$a = b^d \xrightarrow[\text{caz1}]{} T(n) = O(n \log n)$

Ex. Quick Sort

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

$a = 2$, $b = 2$, $d = 1$

$a = b^d \xrightarrow[\text{caz1}]{} T(n) = O(n \log n)$

Ex. Înmulțirea a două nr. de lungime n

→ direct $O(n^2)$

→ Divide et Impera

$x = \boxed{x_L | x_R} = 2^{\frac{n}{2}} \cdot x_L + x_R$

$y = \boxed{y_L | y_R} = 2^{\frac{n}{2}} \cdot y_L + y_R$

$x \cdot y = 2^n x_L y_L + 2^{\frac{n}{2}}(x_L y_R + x_R y_L) + x_R y_R$

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

$a = 4$, $b = 2$, $d = 1$

$a > b^d \xrightarrow[\text{caz3}]{} T(n) = O(n^{\log_2 4}) = O(n^2)$

→ de la 4 apeluri recursive la 3

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

$a = 3$, $b = 2$, $d = 1$

$a > b^d \xrightarrow[\text{caz3}]{} T(n) = O(n^{\log_2 3}) = O(n^{1,59})$

Tip	Timp								Spațiu
	Mediu				Rău				Rău
	Acces	Căutare	Inserție	Ștergere	Acces	Căutare	Inserție	Ștergere	
Vector	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Stivă	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Coada	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Listă simplă	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Listă dublă	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Skip-list	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$\Theta(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n\log n)$
Hashtable	N/A	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	N/A	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Arbore binar de căutare	$\Theta(\log(n))$	$\log(n)$	$\log(n)$	$\log(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Cartesian tree	N/A	$\log(n)$	$\log(n)$	$\log(n)$	N/A	$O(n)$	$O(n)$	$O(n)$	$O(n)$
B-Tree	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$O(n)$
Red-Black Tree	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$O(n)$
Splay tree	N/A	$\log(n)$	$\log(n)$	$\log(n)$	N/A	$\log(n)$	$\log(n)$	$\log(n)$	$O(n)$
AVL tree	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	$O(n)$
KD tree	$\log(n)$	$\log(n)$	$\log(n)$	$\log(n)$	n	n	n	n	$O(n)$

Algoritm	Timp			Spațiu
	Bun	Mediu	Rău	Rău
Quicksort	$n\log n$	$n\log n$	n^2	$\log n$
Mergesort	$n\log n$	$n\log n$	$n\log n$	n
Timsort	n	$n\log n$	$n\log n$	n
Heapsort	$n\log n$	$n\log n$	$n\log n$	1
Bubble Sort	n	n^2	n^2	1
Inserție	n	n^2	n^2	1
Selecție	n^2	n^2	n^2	1
Tree Sort	$n\log n$	$n\log n$	n^2	n
Shell sort	$n\log n$	$(n\log n)^2$	$n(\log^2(n))$	1
Bucket sort	$n+k$	$n+k$	n^2	n
Radix Sort	nk	nk	nk	$n+k$
Counting	$n+k$	$n+k$	$n+k$	k
Cubesort	n	$n\log n$	$n\log n$	n

Aplicații → Teorema Master

1. $T(n) = 3T(n/2) + n^2$

   $a = 3$, $b = 2$, $d = 2$

   $a < b^d \Rightarrow T(n) = O(n^2)$ (caz 2)

2. $T(n) = 4T(n/2) + n^2$

   $a = 4$, $b = 2$, $d = 2$

   $a = b^d \Rightarrow T(n) = O(n^2 \log n)$ (caz 1)

3. $T(n) = T(n/2) + 2^n$

   $2^n \in \Omega(n)$

   $a = 1$, $b = 2$, $d = 1$

   $a < b^d \Rightarrow T(n) = O(2^n)$ (caz 2)

Met:

$T(n) = a\,T(n/b) + f(n)$ $a \geq 1, b > 1$

1. $f(n) = O(n^{\log_b a - \varepsilon})$, $\varepsilon > 0 \Rightarrow T(n) = \Theta(n^{\log_b a})$
2. $f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0 \Rightarrow T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$, $\varepsilon > 0 \Rightarrow T(n) = \Theta(f(n))$

5. $T(n) = 16T(n/4) + n$

   $a = 16$, $b = 4$, $d = 1$

   $a > b^d \Rightarrow T(n) = O(n^2)$

6. $T(n) = 2T(n/2) + n\log n$

   $a = 2$, $b = 2$, $d = 2$

   $a < b^d$ ~~$\Rightarrow T(n) = O(n^2)$~~

   $n\log n \in \Theta(n^{\log_b a} \log n) \Rightarrow T(n) = O(n \log^2 n)$

Sortări

→ **interne**
- în aceleași locații (nr. minim de locații suplimentare)
- operația de bază: compararea cheilor

∘ inserție directă

- pasul 1: se ia $A[2]$ și se inserează la locul ei în vectorul de lungime 1, $A[1]$
  ⇓
  vector sortat de lungime 2
- pasul i (pasă):
  $A[i:2]$ → $\underbrace{A[1] \dots A[i]}_{\text{destinație}}$ sortat crescător
  $\underbrace{A[i+1], \dots, A[n]}_{\text{sursă}}$ nesortat

  se ia $A[i+1]$ → se caută de la dr. la stg
  se mută componentele mai mari la dreapta
  se inserează

C_i - nr. de comparații efectuat

$C_i = max = i+1$ (val e minimă)

$C_i = minim = 1$ (val e maximă)

$C_i = medie = i/2$

M_i - mutări la pasul i

$M_i = C_i + 1$

În total:

$\boxed{C_{min}} = \underbrace{1}_{pasul 1} + \underbrace{1}_{pasul 2} + \dots + \underbrace{1}_{pasul\ n-1} = n-1$

Algoritm

$$M_{min} = \underset{p1}{2} + \underset{p2}{2} + \dots + \underset{p\,n-1}{2} = 2(n-1)$$

$$C_{max} = 2 + 3 + \dots + n = \frac{n(n+1)}{2} - 1$$

$$M_{max} = 3 + 4 + \dots + n+1 = \frac{n(n+1)}{2} - 1 + (n-1) = \frac{n^2+3n-4}{2}$$

$$C_{mediu} = \frac{1}{2}(2+3+\dots+n) = \frac{1}{2} C_{max} = \frac{n(n+1)}{4} - \frac{1}{2} =$$

$$= \frac{n^2+n-2}{4}$$

$$M_{mediu} = C_{mediu} + 2(n-1) = \frac{n^2+n-2}{4} - 2(n-1) =$$

$$= \frac{n^2+9n-10}{4}$$

$$C_{min} = n-1 \le T(n) \le \frac{n^2+n-2}{2} = C_{max}$$

$$C_{min} + M_{min} \le T(n) \le C_{max} + M_{max}$$

$$3(n-1) \le T(n) \le n^2+2n-3 \quad \left(\frac{1}{2}n^2 + \frac{3}{2}n - 2\right)$$

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

$$\boxed{T(n) = O(n^2)}$$

$$3(n-1) \le T(n)$$

$$\boxed{T(n) = \Omega(n)}$$

$$T(n) = a\, C_{max} + b \quad (\exists\ a, b > 0)$$

$$\boxed{T(n) = \Theta(n^2)}$$

La pasul $i \rightarrow i+1$ elemente sortate.

- selecție directă a minimului – nu e stabil

la pasul i (pasă): A : 2 – A[1 ... i-1] minime în pe locul lor
dimensiunea crește cu 1

A[i ... n] pe care se efectuează o pasă
↓
se caută minimul → se interschimbă cu A[i]

După n-1 pași → A e sortat.

$C_i(n) = n - i$

$C = (n-1) + (n-2) + \dots + 1 = \frac{n(n-1)}{2}$

fiecare interschimbare ⇒ 3 deplasări

$M_{min} = 3 + 3 + \dots + 3 = 3(n-1)$

$M_{max} = [(n-1)+3] + [(n-2)+3] + \dots + [1+3] =$
$= (n-1) + (n-2) + \dots + 1 + 3(n-1) =$
$= \frac{n(n-1)}{2} + 3(n-1)$

$M_{mediu} = n(\ln \gamma + e)$???

$T(n) = \#C(A) + \#M \le \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + 3(n-1)$

$T(n) = O(n^2)$

- interschimbare directă (Bubblesort) – stabil

se parcurge de la dreapta la stânga
se compară A[i-1] și A[i]
dacă $A[i-1] \le A[i]$ se lasă pe loc

la pasul i:
+1 ← A[1 ... i-1] ↔ cele mai mici valori la locul lor
A[i ... n] ↔ se împinge pe poziția A[i]

după n-1 pași → A e sortat

la pasul i ⇒ $C_i = n - 1$ comparații

$C = (n-1) + (n-2) + \dots + 1 = \frac{n(n-1)}{2}$

$M_{min} = 0$

$M_{max} = (n-1) \cdot 3 + (n-2) \cdot 3 + \dots + 1 \cdot 3 = \frac{3n(n-1)}{2}$

$M_{mediu} = \frac{n-1}{2} \cdot 3 + \frac{n-2}{2} \cdot 3 + \dots + \frac{1}{2} \cdot 3 = \frac{3n(n-1)}{4}$

Pentru fiecare i : $\Rightarrow$ 1 comparație

$\sum_{i=2}^{n} 1 = n-1$

comparații $n-1 \le T_C(n) \le \frac{n(n-1)}{2}$

$\rightarrow$ maxim 2 deplasări

$0 \le T_i \le 2(n-1)$

deplasări: $0 \le T_D(n) \le n(n-1)$

complexitate

totală: $n-1 \le T(n) = \cancel{T(1+n)}\ T_C + T_D \le \frac{n(n+1)}{2} + n(n-1)$

$\Downarrow$

$T(n) = O(n^2)$

cel mai favorabil caz : $1, 2, 3, \dots, n$

cel mai nefavorabil caz : $n, n-1, \dots, 1$

- mergesort - stabil

$$T(n) = \begin{cases} \Theta(1), & n = 1 \\ 2T(n/2) + \Theta(n), & n > 1 \end{cases}$$

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

$$T(n) = \begin{cases} 1, & n = 1 \\ T(\lceil \frac{n}{2} \rceil) + T(\lfloor \frac{n}{2} \rfloor) + n, & n > 1 \end{cases}$$

• quicksort

→ partiția Hoare

13 19 9 5 12 $\boxed{8}$ 7 4 11 2 6 21

L ... R

2 m interschimbări

→ partiția Lomuto

m+1 interschimbări

13 19 9 5 12 21 7 4 11 2 6 $\boxed{8}$

L ... R

5 19 9 13 12 21 7 4 11 2 6 $\boxed{8}$

5 7 9 13 12 21 19 4 11 2 6 $\boxed{8}$

5 7 4 13 12 21 19 9 11 2 6 $\boxed{8}$

5 7 4 2 12 21 19 9 11 13 6 $\boxed{8}$

5 7 4 2 6 21 19 9 11 13 12 $\boxed{8}$

5 7 4 2 6 $\boxed{8}$ 19 9 11 13 12 21

• quicksort

→ nerandomizat

$$T(n) = \sum_{p=1}^{n} \frac{1}{n}\left(T(p-1) + T(n-p)\right) + n-1$$

pivotul p e ales cu aceeași probabilitate

n−1 comparații necesare partiționării

H_n − număr armonic: $H_n = \sum_{i=1}^{n} \frac{1}{i} \cong \ln n$

$$T(n) = \frac{2}{n}\sum_{p=1}^{n} T(p-1) + n-1$$

$$nT(n) = 2\sum_{p=1}^{n} T(p-1) + n(n-1)$$

$$(n-1)T(n-1) = 2\sum_{p=1}^{n-1} T(p-1) + (n-1)(n-2) \quad (-)$$

$$nT(n)-(n-1)T(n-1)=2T(n-1)+2(n-1)$$

$$\Rightarrow \frac{T(n)}{n+1}=\frac{T(n-1)}{n}+\frac{2(n-1)}{n(n+1)}$$

$$a_n=\frac{T(n)}{n+1}$$

$$a_n=\cancel{a_{n-1}}+\frac{2(n-1)}{n(n+1)}$$

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

$$\text{— — — — — —}$$

$$\cancel{a_1}=T(1)=0$$

$$(+)$$

$$a_n=\sum_{i=1}^{n}\frac{2(i-1)}{i(i+1)}.$$

$$a_n\simeq 2\sum_{i=1}^{n}\frac{1}{i+1}$$

$$a_n\simeq 2\ln n$$

$\Rightarrow$ Deci $T(n)=(n+1)a_n\simeq 2(n+1)\ln n \simeq 1{,}38\, n\lg n (\ln n)$

$\rightarrow$ randomizat

- aceeași probabilitate

① Cazul cel mai defavorabil (merge la amândouă) split – cazul "worst": $\Theta(n^2)$

$$\boxed{T(n)=\max_{0\le q\le n-1}\left(T(q)+T(n-q-1)\right)+\Theta(n)}$$

$\exists\, c$ a.î. $T(n)\le cn^2$

$$T(n)\le \max_{0\le q\le n-1}\left(cq^2+c(n-q-1)^2\right)+\Theta(n)=$$

$$=c\max_{0\le q\le n-1}\left(q^2+(n-q-1)^2\right)+\Theta(n)$$

$\max (q^2+(n-q-1)^2)$ apare în 0 și $n-1$

(derivata a doua în raport cu q e >0)

$$\max_{0\le q\le n-1} (q^2+(n-q-1)^2) \le (n-1)^2 = n^2-2n+1$$

Deci $T(n) \le cn^2 - c(2n-1) + \Theta(n) \le cn^2$

$\Downarrow$

$T(n) = O(n^2)$

- Fie $z_1, z_2, \dots, z_n$ — elementele din A

$Z_{ij} = \{z_i, z_{i+1}, \dots, z_j\}$

Fie $X_{ij} = I\{z_i$ este comparat cu $z_j\}$

Numărul total de comparații:

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

$$E[X] = E\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right] =$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] =$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Pr\{z_i \text{ este comparat cu } z_j\}$$

~~Dacă~~ Fie x pivot a.î. $z_i < x < z_j$

nu se compară

Se compară $\iff$ primul element pivot e ~~în~~ z_i sau z_j

$$Pr = \frac{1}{j-i+1}$$

$Pr\{z_i$ este comparat cu $z_j\} = Pr\{z_i$ sau z_j este ales prima oară pivot din $Z_{ij}\} = Pr\{z_i$ e ales primul$\}$ $+ Pr\{z_j$ e ales primul$\} = \dfrac{1}{j-i+1} + \dfrac{1}{j-i+1} = \dfrac{2}{j-i+1}$

$$E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

Fie $k = j - i$

$$E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-1} \frac{2}{k+1}$$

$$\wedge \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{2}{k} = \sum_{i=1}^{n-1} O(\lg n) =$$

$$= O(n \lg n)$$

Bubblesort – după pasul k → ultimele/primele k elemente ale șirului sunt sortate

Insertion sort – — — → primele $k+1$ sunt sortate

Selection sort – — — → primele/ultimele k elemente sunt sortate

Quick sort – — — , având p pivot → el mai mici, p, elemente mai mari

Structuri de date

Arbori

→ traversări:

- preordine (RSD)
- inordine (SRD)
- postordine (SDR)

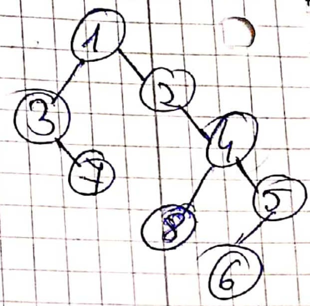
• ștergere: – se caută elementul
- dacă are 1 fiu → se șterge direct
- altfel, se înlocuiește cu predecesorul/succesorul din SRD → se continuă ștergerea

- reconstructie

stg. dr.

• SRD: 3 7 $\boxed{1}$ 2 8 4 6 5

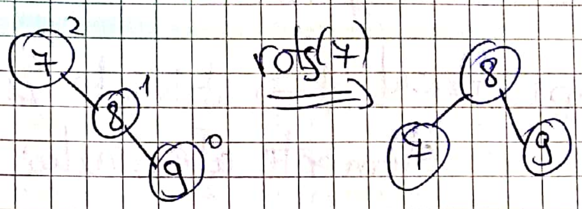
• RSD: $\boxed{1}$ 3 7 2 4 8 5 6



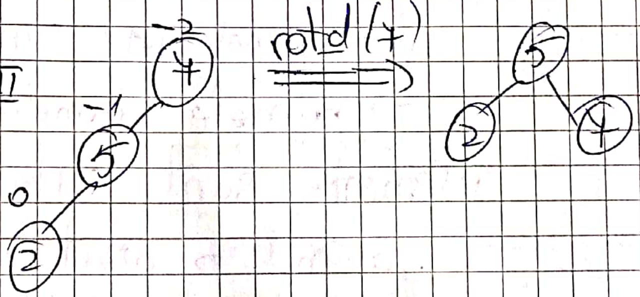
AVL

echilibrari

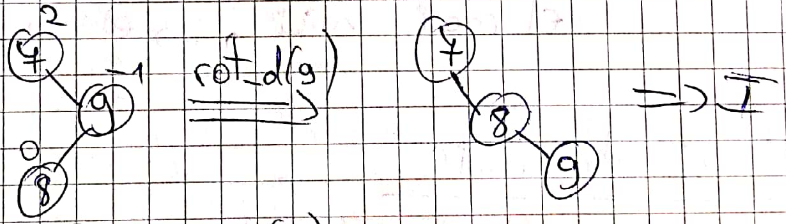
I



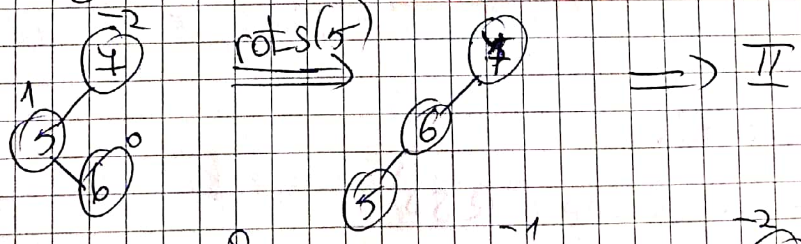
II



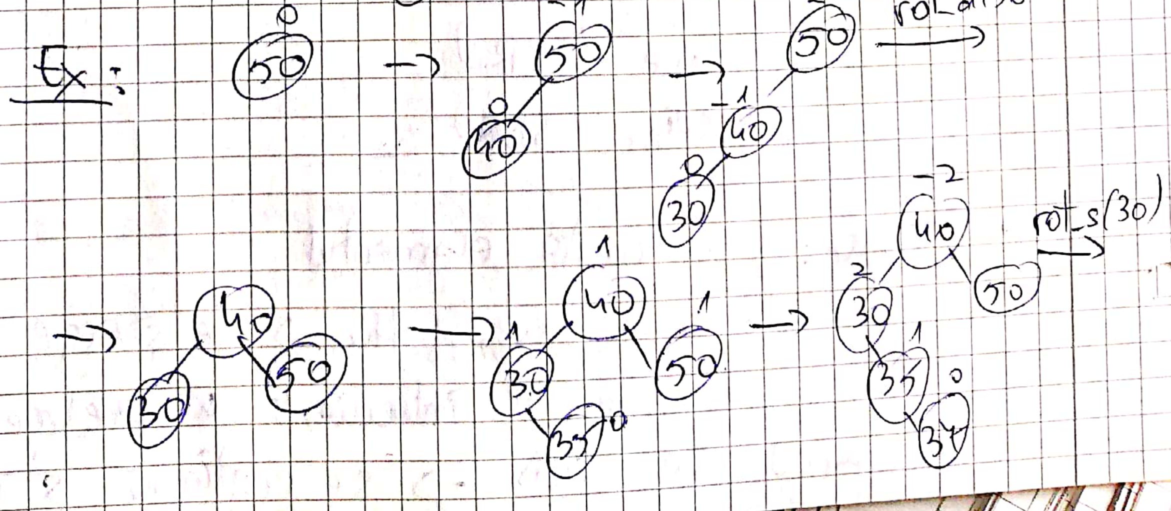
III

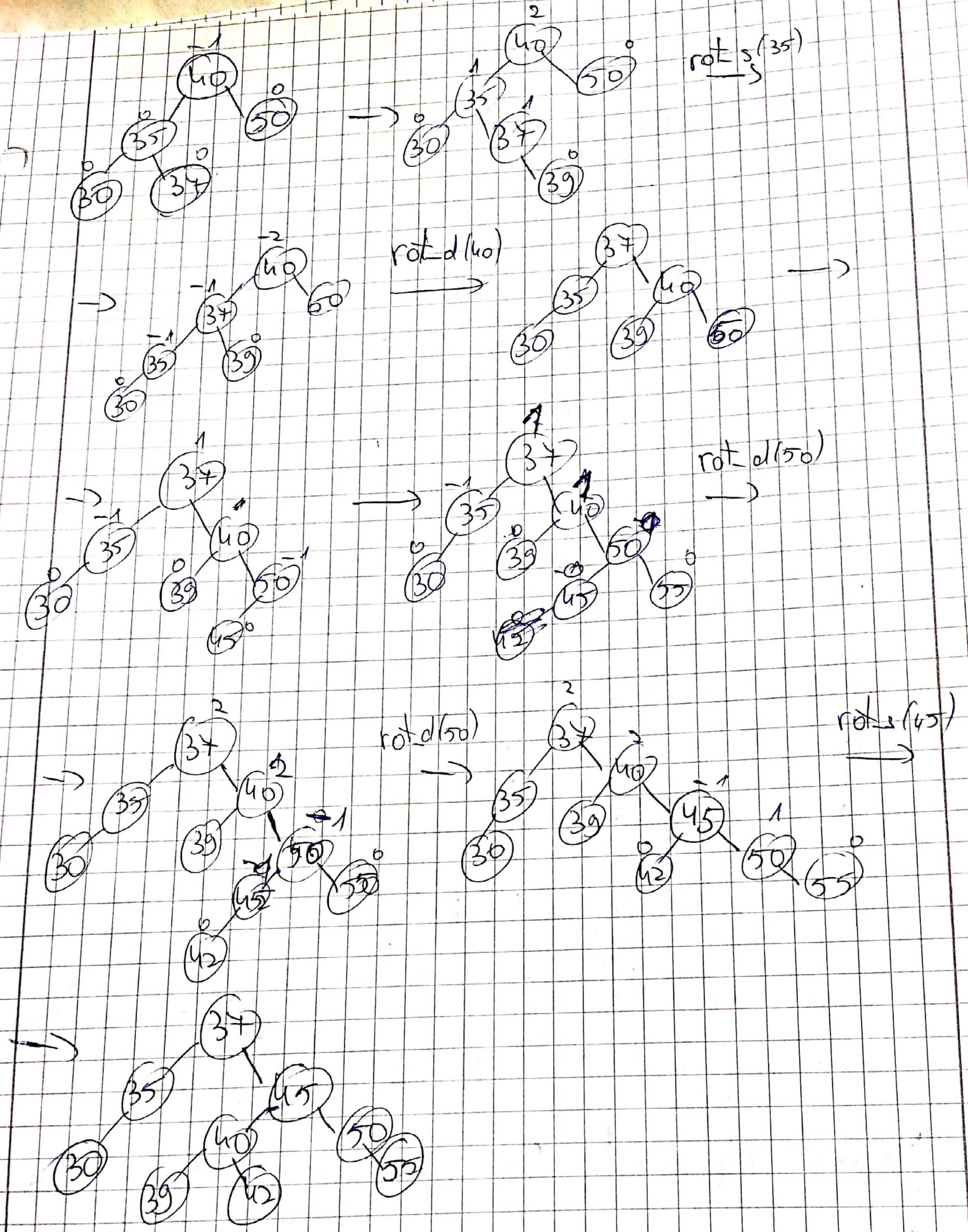


IV



Ex:





Arbori binari stricți

Prop. 1 $N_E = N_I + 1$

Dem: $2N_I = N_I + N_E - 1$

$N_E = N_I + 1$

- $L_E = \sum_{x \in E} l(r, x)$
- $L_I = \sum_{y \in I} l(r, y)$

Prop. 2 $L_E = L_I + 2N_I$

Dem. Pp. $L_E = L_I + 2N_I$

Fie T arbore binar strict cu n noduri, $n<m$

$\Downarrow$

T^s, T^d

$N_I^s, N_E^s, L_I^s, L_E^s$; $N_I^d, N_E^d, L_I^d, L_E^d$

1) $N_I = N_I^s + N_I^d + 1$

2) $N_E = N_E^s + N_E^d$

3) $L_E = L_E^s + N_E^s + L_E^d + N_E^d$

4) $L_I = L_I^s + N_I^s + L_I^d + N_I^d$

5) $L_E^s = L_I^s + 2N_I^s$

6) $L_E^d = L_I^d + 2N_I^d$

$L_E = L_E^s + N_E^s + L_E^d + N_E^d =$

$= L_I^s + N_I^s + N_E^s + N_E^d + L_I^d + N_I^d + N_I^d =$

$= (L_I^s + N_I^s + L_I^d + N_I^d) + (2N_I^s + 1) + (2N_I^d + 1)$

$L_E = L_I + 2(N_I^s + N_I^d + 1) = L_I + 2N_I$

Prop. 3 $N_E \leq 2^d$ (d adâncime) $\Rightarrow d \geq \log_2 N_E$

Prop. 4 2 arbori cu N_E nr. frunze

$\Downarrow$

L_E minimă $\Leftrightarrow$ frunze pe cel mult 2 niveluri adiacente

Prop. 5 $L_E^{min} = l\lfloor \lg l \rfloor + 2(l - 2^{\lfloor \lg l \rfloor})$, l frunze

Prop. 6 $L_E^{medie} \geq \lfloor \lg l \rfloor$

Nr. de frunze la nivelul $d-1$: $Y = 2^d - N_E$

Nr. de frunze la nivelul d: $2X = 2N_E - 2^d$

$\overline{X+Y}$

Teorema AVL

Fie T un arbore binar strict și echilibrat AVL, cu n noduri interne. Fie $h(T)$ înălțimea lui. Avem

$$\log_2(n+1) \le h(T) \le 1{,}440\log_2(n+2) - 0{,}328$$

Echivalent sunt satisfăcute următoarele inegalități

(1) $h(T) \geq \log_2(n+1)$

(2) $h(T) \le \frac{1}{\log_2 \phi} \log_2(n+2) + \frac{\log_2 5}{2\log_2 \phi} - 2$, $\phi = (1+\sqrt{5})/2$.

Arbori binari stricți $\to$ căutare binară

$C_n = \left(1 + \frac{1}{n}\right)C'_n - 1$

$C_n = \frac{LI(T)}{n} + 1$

$C'_n = \frac{LE(T)}{n+1}$

Căutare maxim: $O(1)$

Extragere maxim: $O(\log(n))$

increase key: $O(\log(n))$

Insert: $O(\log(n))$

Ștergere: $O(\log(n))$

Merge: $O(m+n)$

Heap

– înălțime $\Theta(\lg n)$

heapsort $\to$ după n pași iterativi, A e sortat

$O(n\lg n)$

Teorema limitei inferioare

Orice algoritm de sortare a n chei, algoritm bazat pe comparații între chei, va face cel puțin:

$\lceil \log_2 n! \rceil$ comparații în cazul cel mai nefavorabil

$\lfloor \log_2 n! \rfloor$ comparații în cazul mediu

Grafuri

$G(V, E)$

Tip		complexitate timp				
	storage	Add Vertex	Add edge	R.V.	R.E.	Query
Listă de adiacență	$O(V+E)$	$O(1)$	$O(1)$	$O(V+E)$	$O(E)$	$O(V)$
Listă de incidență	$O(V+E)$	$O(1)$	$O(1)$	$O(E)$	$O(E)$	$O(E)$
Matrice de adiacență	$O(V^2)$	$O(V^2)$	$O(1)$	$O(V^2)$	$O(1)$	$O(1)$
Matrice de incidență	$O(V \cdot E)$	$O(V \cdot E)$	$O(V \cdot E)$	$O(V \cdot E)$	$O(V \cdot E)$	$O(E)$

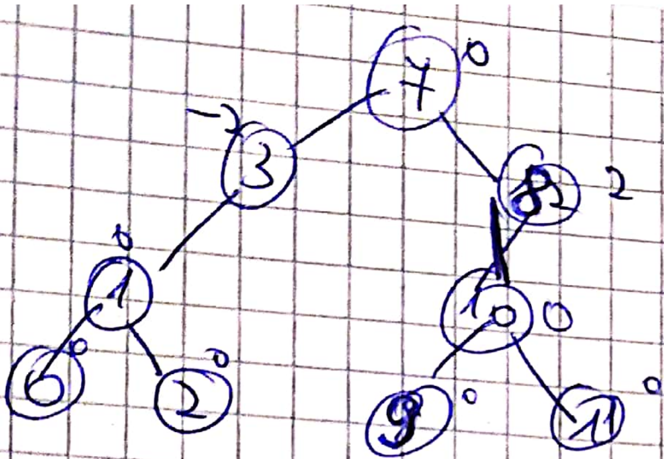
$n < m$

BFS: → cu lista: $O(E+V)$
→ cu matrice: $O(V^2)$

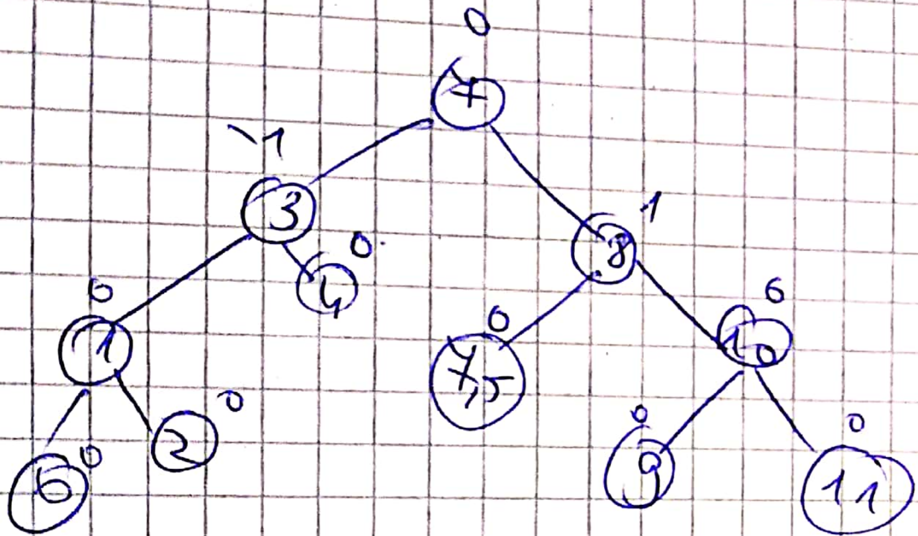
DFS: → cu lista: $O(E+V)$
→ cu matrice: $O(V^2)$

Sortare topologica: $O(E+V)$

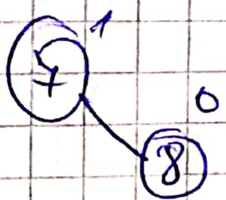
d-s



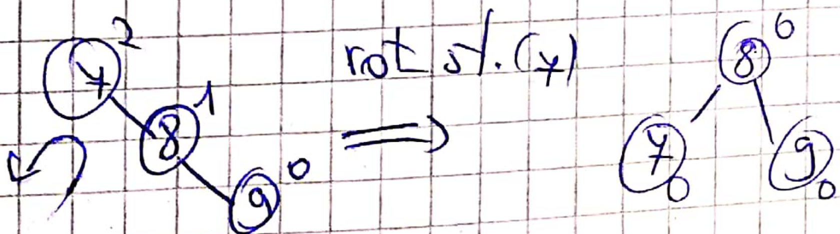
Nu e AVL!



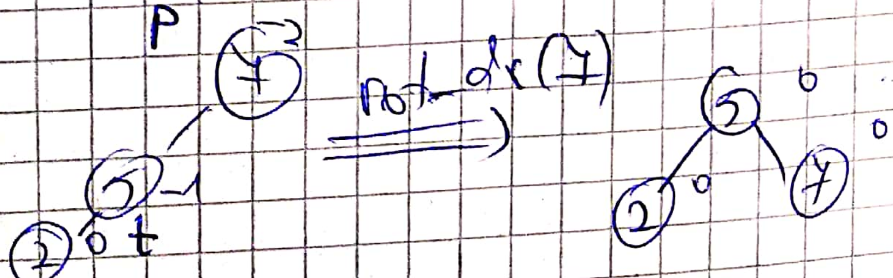
E AVL!



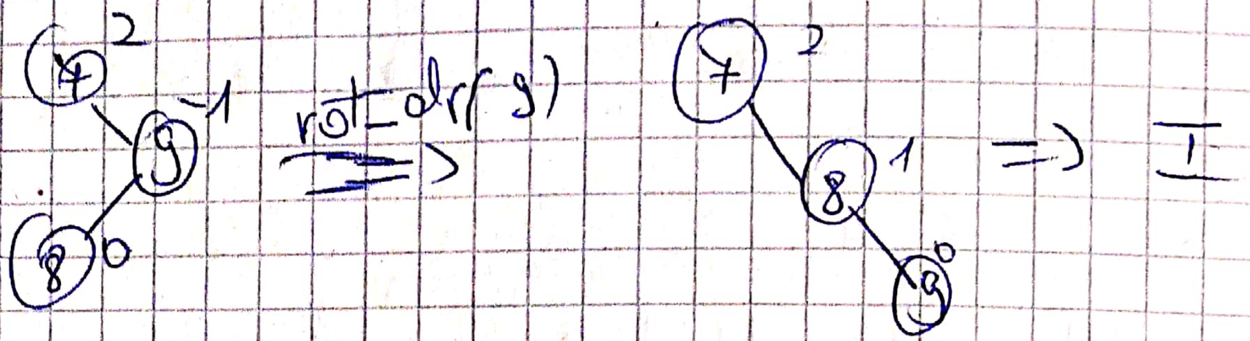
I



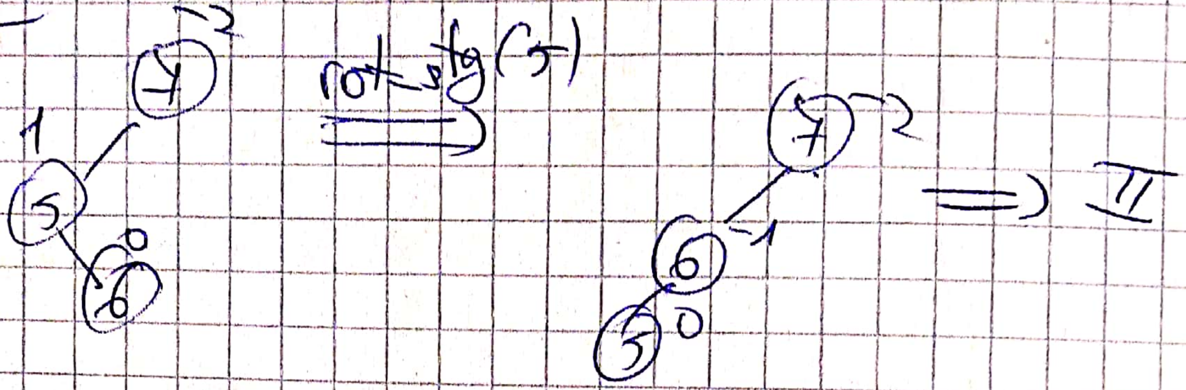
II



III



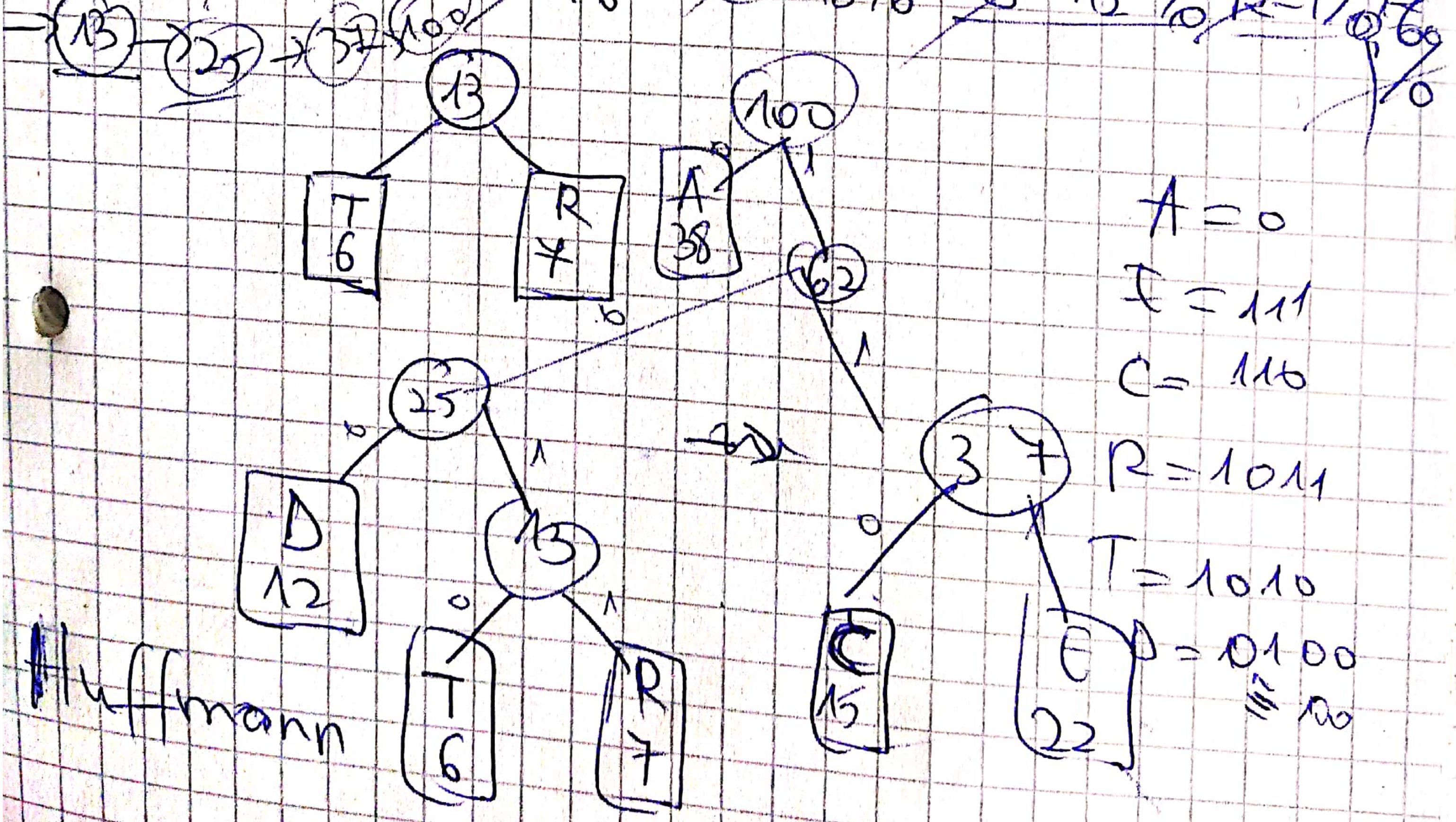
IV



Ex 8 5 1 6 7 10 9

Probleme

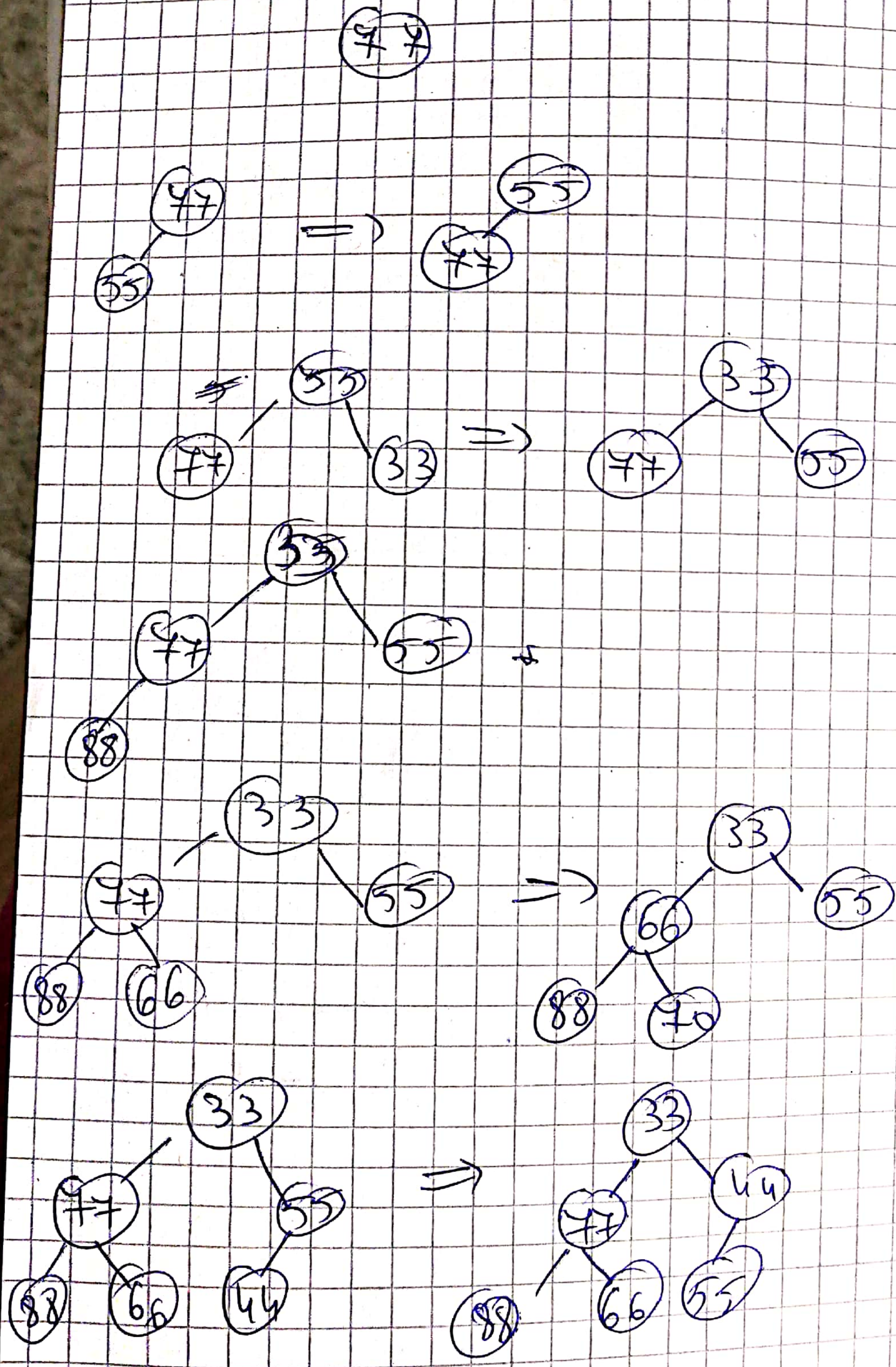
A - 38% E - 22% C - 15% D - 12% R - 7% T - 6%



A = 0
E = 111
C = 110
R = 1011
T = 1010
D = 0100

Huffmann

16 77 55 33 88 66 44



ASD - laborator

Sortări: 7, 5, 2, 4, 3, 1

→ interschimbare directă • bubblesort

• swap

I 7, 5, 2, 4, 3, 1

5, 7, 2, 4, 3, 1

2, 7, 5, 4, 3, 1

2, 7, 5, 4, 3, 1

1, 7, 5, 4, 3, 2

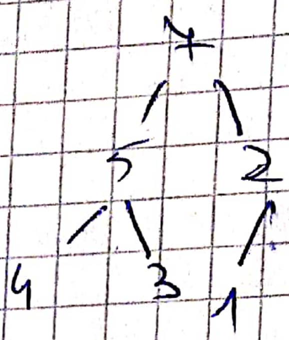
II 1, 7, 5, 4, 3, 2

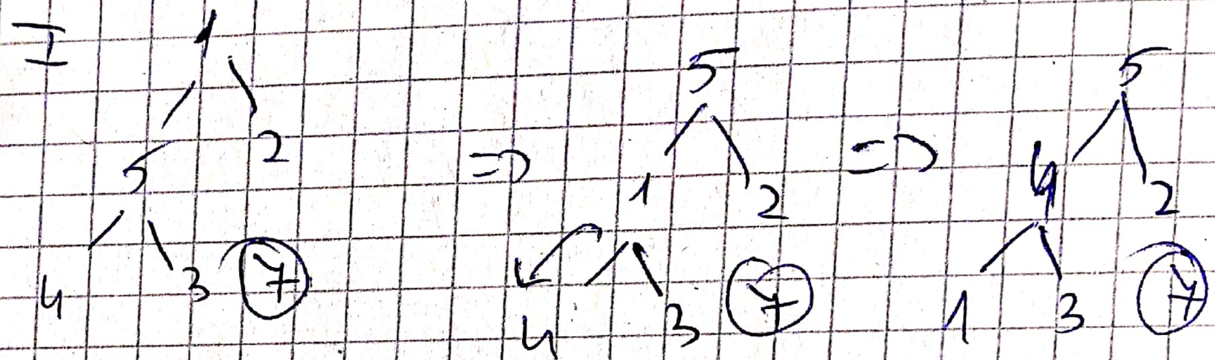
1, 5, 7, 4, 3, 2

1, 2, 7, 5, 4, 3

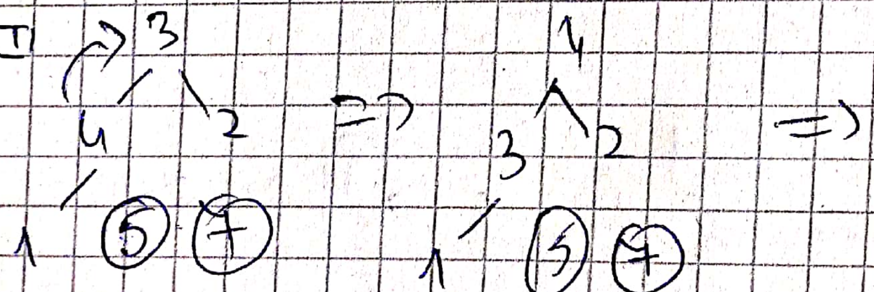
4) heapsort (ordonare crescătoare) → max ansamblu + decapitări

(dacă nu este max, îl creăm)





5 4 2 1 3 $\boxed{7}$



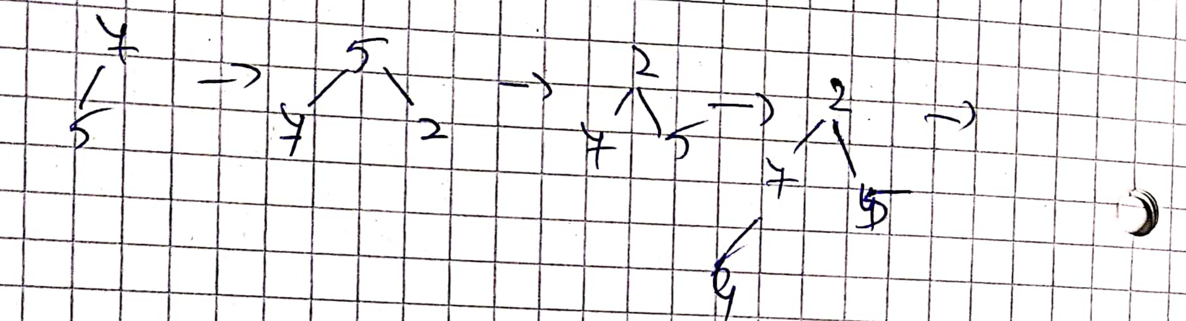
4 3 2 1 $\boxed{5}\boxed{7}$

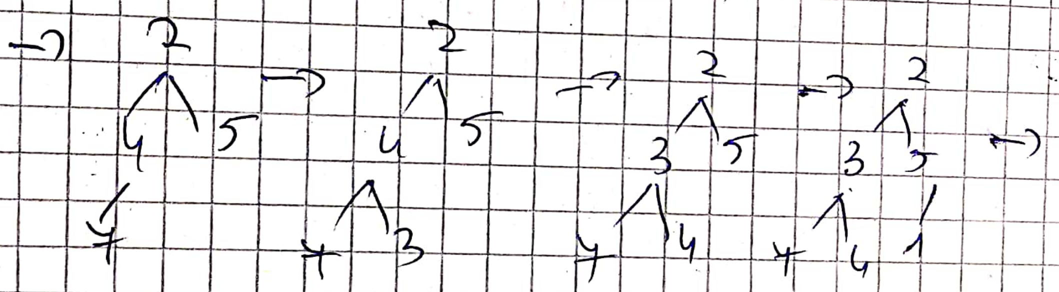
după x pași de heapsort, cele mai mari x elemente sunt pe ultimele poziții, ordonate crescător

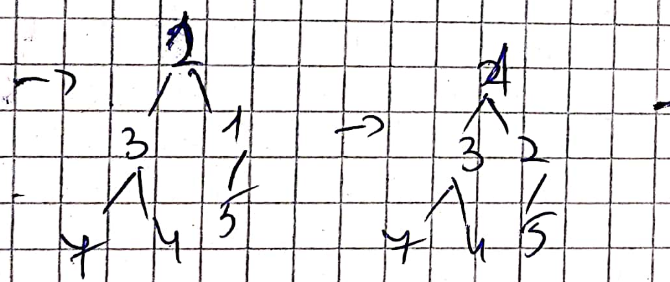
• heapsort (ordonare descrescătoare

→ min ansamblu + decapitări
(dacă nu e min, îl creăm)

I var iterativă pt. creare min-ansamblu

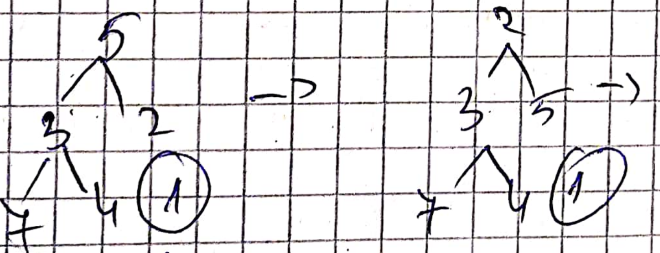


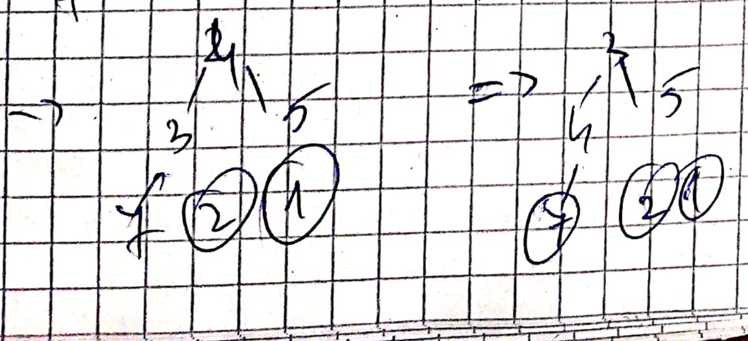




→ $\boxed{1}$, 3, 2, 7, 4, 5.

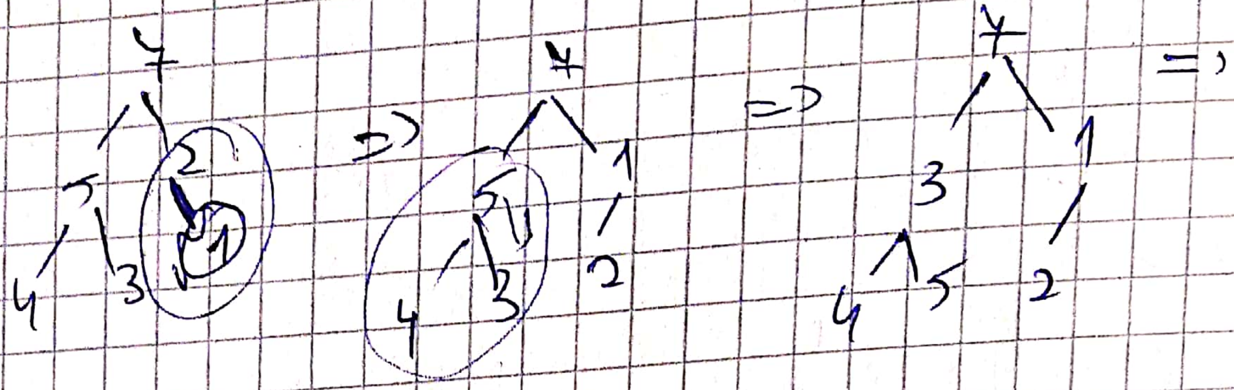
2, 3, 5, 7, 4, $\boxed{1}$

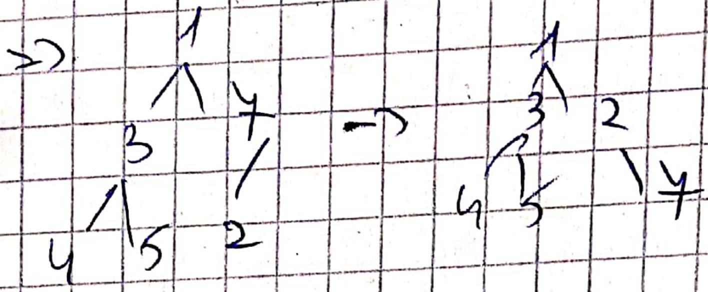


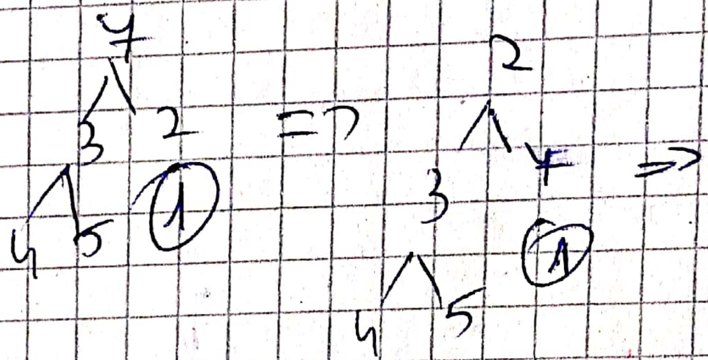


3, 4, 5, 7, $\boxed{2, 1}$

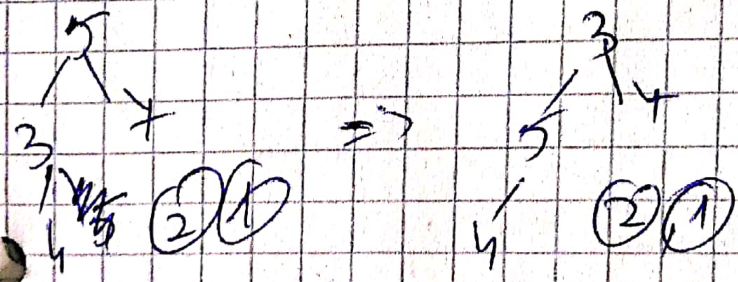
II Var recursivă pt. creare min ansamblu







=> 2, 3, 7, 4, 5, $\boxed{1}$



=> 3, 5, 7, 4, $\boxed{2}$, $\boxed{1}$

Sortări

-> quicksort

- partiția Hoare

x = 2

7, 5, 2, 4, 3, 1

1, 5, 2, 4, 3, 7

1 2 5 4 3 7

divide

5, 4, 3, 7

x = 4

5, 4, 3 7

3 4 5 7

- partiția 2

x = 7

$\boxed{7}$, 5, 2, 4, 3, 1

1, 5, 2, 4, 3, $\boxed{7}$

$\boxed{1}$, 5, 2, 4, 3, $\boxed{7}$

$\boxed{5}$, 2, 4, 3

3, 2, 4, $\boxed{5}$

3, 2, 4

:

-> Lomuto

7, 5, 2, 4, 3, $\boxed{1}$

$\boxed{1}$, 5, 2, 4, 3, 7

5, 2, 4, 3, $\boxed{7}$

5, 2, 4, $\boxed{3}$

2, 5 4 $\boxed{3}$

2 $\boxed{3}$ 4 5

→ shellsort

7 5 2 4 3 1

$h_{n+1} = \frac{h_n}{2}$

$h_1 = n/2$

$h_1 = 3$

I 7 4 ⇒ 4 7

5 3 ⇒ 3 5

2 1 ⇒ 1 2

4, 3, 1, 7, 5, 2

$h_2 = 3/2 = 1$

II (1) 3, 4, 1, 7, 5, 2

(2) 1, 3, 4, 7, 5, 2

(3) 1, 3, 4, 7, 5, 2

(4) 1, 3, 4, 5, 7, 2

(5) 1, 2, 3, 4, 5, 7

h = 3

7 4 → 4 7

5 3 → 3 5

2 1 → 1 2

⇒ 4, 3, 1, 7, 5, 2

h = 2

4 1 5 ⇒ 1 4 5

3 7 2 ⇒ 2 3 7

1 2 4 3 5 7

h = 1

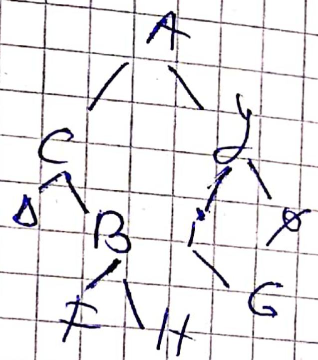
1, 2, 4, 3, 5, 7

1, 2, 3, 4, 5, 7

RSD : A C D B F H J i G

SRD : D C F B H A i G J

~~A~~



RSD : C D B F H

SRD : D C F B H

RSD : B F H

SRD : F B H

RSD : J i G

SRD : i G J

RSD : i G

SRD : i G

SDR : D F H B C G i J $\boxed{A}$

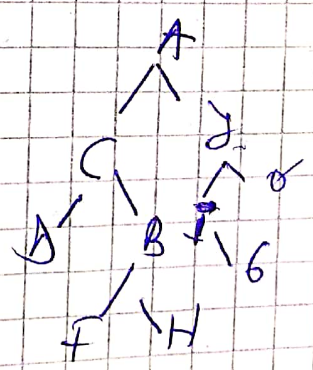
SRD : D C F B H $\boxed{A}$ i G J

SDR : D F H B $\boxed{C}$

SRD : D C F B H

SDR : D F H B

SRD : F B H



SDR: Gi$\boxed{j}$

SRD: iG$\boxed{j}$

SDR: G$\boxed{i}$

SRD: $\boxed{i}$G

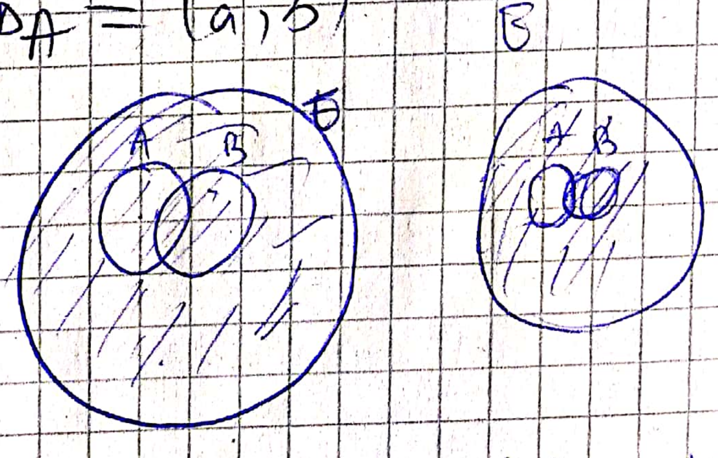
$R \subseteq A \times B$ $\quad \Delta_A \subseteq A \times A$ $\quad \Delta_B \subseteq B * B$

(a,b) $\quad Q = (b,c)$ $\quad (a,a)$ $\quad (b,b)$

$Q \circ R = \{(a,c) / \exists\, b \in B$ a.î. $(a,b) \in R$ și $(b,c) \in Q\}$

$\Delta_B \circ R = \{(a,b) / \exists$

$R \circ \Delta_A = (a,b)$



$x \in C_E(A \cup B) \Rightarrow x \in E \setminus (A \cup B) \Rightarrow x \in E \setminus A \cap E \setminus B$

$x \in C_E A \cap C_E B \Rightarrow x \in (E \setminus A) \cap (E \setminus B)$

$\Downarrow$

$x \in E \setminus A$ și $x \in E \setminus B$

$\Downarrow$

$x \in E \setminus (A \cup B)$