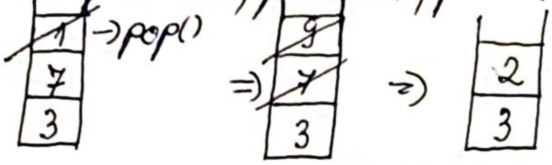
= STIVE. COZI =

1) STIVĂ

• <u>LIFO</u> = 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

• <u>FIFO</u> = 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

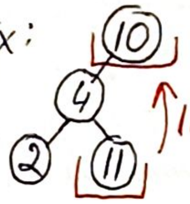
IN: 2 9 ~~3~~ ~~7~~ :OUT

IN: 2 9 : OUT

= ARBORE BINAR DE CĂUTARE =

<u>DEF:</u> 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.

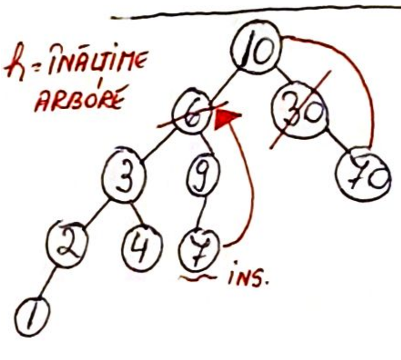
Ex:



11 > 10 ⇒ NU E A.B.C.

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

h = ÎNĂLȚIME ARBORE



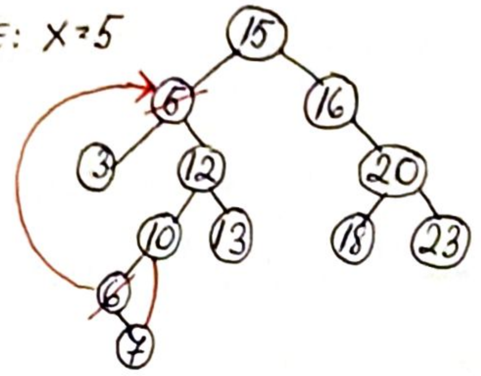
1) <u>INSERȚIE</u>: pt 7 $\boxed{\Rightarrow \in O(h)}$
7 < 10 → stânga; 7 > 6 → dreapta; 7 < 9 → stânga

2) <u>CĂUTARE</u>: pt. 2 ⇒ $\boxed{\in O(h)}$; h = ÎNĂLȚIME ARB.
2 < 10 → st; 2 < 6 → st; 2 < 3 → st; 2 = 2 → GĂSIT

3) <u>ȘTERGERE</u>: 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) } ⇒ $\boxed{\in O(h)}$

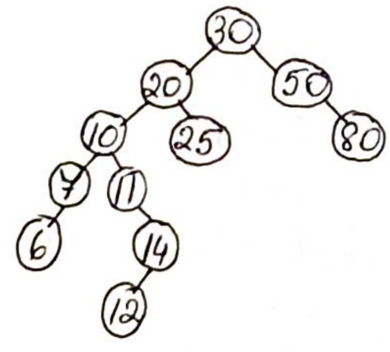
5) SUCCESOR: CEL MAI MIC NOD MAI MARE DECÂT "X" (EX: SUCCESOR 10 = 12)
a) "X" ARE FIU DR ⇒ SUCCESOR = MIN DIN SUBARBORE DR.
b) "X" NU " —"—" ⇒ " —"—" = CEL MAI JOS "Y" A.Î. "Y" ȘI FIU ST. "Y" SUNT STRĂMOȘI PT. "X".

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

→ $\boxed{\in O(h)}$

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

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

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

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

PRIN ABSURD, $(\exists)$ y a.î. y = FIU STÂNG AL SUCC(X).
y = FIU STÂNG SUCC(X) $\Rightarrow$ y < SUCC(X)
y și SUCC(X) SUNT ÎN SUBARBORE DREPT X $\Rightarrow$ y, SUCC(X) > X $\Big\}$ $\Rightarrow$ contradicție (y == SUCC(X))

2) ARBORE BINAR PLIN = $(\forall)$ NOD (CU EXCEPȚIA FRUNZELOR) ARE EXACT 2 FII.
DEM. CĂ $(\forall)$ ARBORE NEPLIN, NU POATE CORESPUNDE UNUI COD OPTIM.

ARB. OPTIM = ARE COSTUL MINIM.

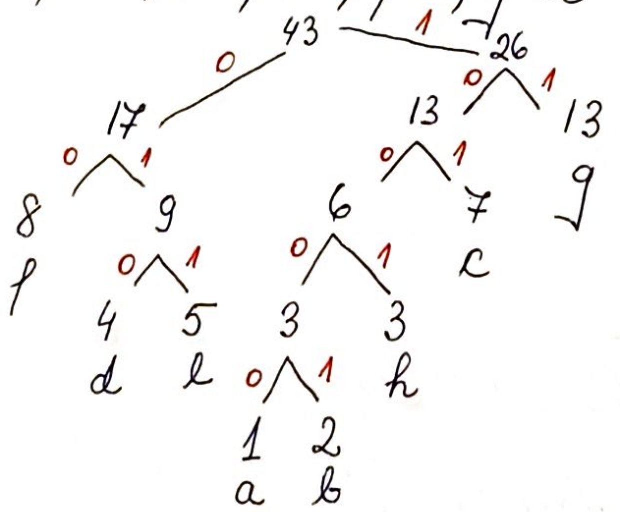
$$\text{COST(ARBORE)} = \sum_{i \in \text{FRUNZĂ}} \text{FRECVENȚĂ}(i) \cdot \text{ADÂNCIME}(i)$$

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

PRESUPUN "T" = ARB. NEPLIN. PRIN ABSURD, E OPTIM..
T = NEPLIN $\Rightarrow$ $(\exists)$ NODUL $v \in T$ CARE ARE UN SINGUR FIU. DACĂ ELIMIN v DIN T, ADÂNCIMEA TUTUROR NODURILOR DIN SUBARBORELE LUI "v" VA SCĂDEA $\Rightarrow$
$\Rightarrow$ cost $(T - \{v\})$ < cost T contradicție (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

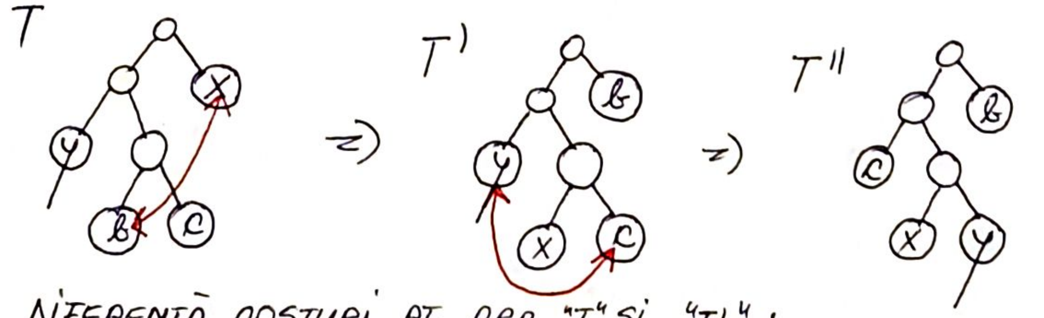
DEMONSTRAȚIE CORECTITUDINE ALGORITM HUFFMAN
METODA GREEDY

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" ∈ 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" $\in C$, 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 $(\exists)$ UN ARB. T'' ALE CĂRUI FRUNZE $\in C'$ 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'.