FIȘĂ DE LUCRU · CLASA A XI-A · FIȘA 7 · ACOPERĂ LECȚIILE XI.13 ȘI XI.20

Arbori și arbori binari

Durată: 100 min Nivel: mate-info

CE VEI ȘTI DUPĂ LECȚIE

- Citești dintr-un vector de tați rădăcina, frunzele, fiii, nivelurile, înălțimea și drumurile.
- Parcurgi un arbore binar (RSD, SRD, SDR, pe niveluri) și îl reconstitui din două parcurgeri.
- Construiești și folosești un arbore binar de căutare: inserare, căutare, inordine sortată.

Ce trebuie să știi (o pagină)

Regula	Exemplu / formulă	Observație
arbore = conex + fără cicluri	n vârfuri ⇒ **n − 1** muchii	pădure cu p arbori: n − p muchii
vectorul de tați: `t[x]` = tatăl lui x	`t[rad] = 0` (exact un 0)	fiii lui x: indicii i cu `t[i] = x`
frunză = nu are fii	nu apare ca **valoare** în `t`	nu „grad 1": rădăcina poate avea un fiu
nivel: rădăcina 0, fiul = tatăl + 1	urci cu `while (t[y]) y = t[y];`	înălțimea = nivelul maxim (în muchii)
din muchii + rădăcină în `t`	DFS din rădăcină: `t[y] = x` pentru `y != t[x]`	
arbore binar: `st[x]`, `dr[x]` (0 = lipsește)	sau `struct nod { int info; nod *st, *dr; };`	
preordine **R**SD, inordine S**R**D, postordine SD**R**	`if (x == 0) return;` la început	pe niveluri: BFS cu coadă
înălțime	`if (x == 0) return -1; return 1 + max(h(st), h(dr));`	frunza are înălțimea 0
înălțime h: noduri ≤ 2^(h+1) − 1, frunze ≤ 2^h	noduri cu 0 sau 2 fii: frunze = interne + 1	înălțimea minimă pentru n noduri: $\lfloor \log_2 n \rfloor$
reconstituire	preordine + inordine sau postordine + inordine	preordine + postordine **nu** ajung
BST: stânga < nod < dreapta, pentru **tot** subarborele	inserezi coborând; `nod *&r`	inordinea dă valorile **sortate**

SFAT

La orice întrebare pe un vector de tați, desenează întâi arborele pe niveluri (rădăcina sus, fiii dedesubt). La parcurgeri, mergi cu degetul pe desen și spune cu voce tare „R, S, D".

Nivelul 1 · Prezice

★ citește și prezice

Arborele **T** are n = 10 și vectorul de tați `t = (5, 5, 0, 3, 3, 4, 1, 1, 6, 6)`.

P1. Pentru T: care este rădăcina? Care sunt frunzele? Completează tabelul. Cât este înălțimea? Care este fratele lui 1? Scrie drumul de la 9 la rădăcină.

vârf	1	2	3	4	5	6	7	8	9	10
număr de fii										
nivel										

P2. Ce returnează `f(3)`, `f(5)`, `f(4)` și `f(7)` pentru T? Spune în cuvinte ce calculează `f`.

```
int f(int x) {
    int s = 0;
    for (int i = 1; i <= n; i++) if (t[i] == x) s += 1 + f(i);
    return s;
}
```

P3. Arborele are muchiile [1,2], [2,3], [2,4], [4,5], [4,6], [6,7]. Scrie vectorul de tați când rădăcina este 4. Apoi scrie-l când rădăcina este 7. Ce poziții s-au schimbat și de ce exact acelea?

P4. Completează, fără desen.

Întrebare	Răspuns	Întrebare	Răspuns
noduri, arbore cu 15 muchii		arbore binar de înălțime 5: noduri maxim / frunze maxim	
muchii, pădure: 20 de vârfuri, 4 arbori		înălțimea minimă a unui arbore binar cu 100 de noduri	
noduri, arbore binar cu 0 sau 2 fii și 10 frunze		muchii de eliminat din K_8 ca să rămână un arbore	
valori 0 în vectorul de tați, 12 noduri		suma gradelor, arbore cu 9 vârfuri	

P5. Arborele binar este dat prin trei vectori (rădăcina este 1). Scrie preordinea, inordinea și postordinea (literele). Care este înălțimea? Câte frunze are? Câte noduri au exact un fiu?

nod	1	2	3	4	5	6	7	8
`v`	M	D	S	B	H	F	X	P
`st`	2	4	0	0	7	8	0	0
`dr`	3	5	6	0	0	0	0	0

P6. Se inserează, în ordine, într-un BST: `40 25 60 10 30 50 70 35 5`. Scrie preordinea, postordinea și parcurgerea pe niveluri. Cât este înălțimea? Ce noduri se vizitează când se caută 33?

P7. Un arbore binar are postordinea `4 7 5 2 6 3 1` și inordinea `4 2 7 5 1 3 6`. Reconstituie-l. Scrie preordinea, înălțimea și frunzele. (Indiciu: rădăcina este **ultima** în postordine.)

P8. Desenează arborele expresiei `(a + b) * (c – d / e)`. Scrie forma în preordine și în postordine. Calculează cu o stivă expresia în postordine `2 4 + 9 6 3 / – *`.

Nivelul 2 · Găsește și repară

★★ găsește și repară

P9. Pentru `t = (2, 0, 2, 1, 4)` programul ar trebui să afișeze nivelurile `1 0 1 2 3`. Ce afișează? Repară mutând un singur rând.

```
int nivel = 0;
for (int i = 1; i <= n; i++) {
    int y = i;
    while (t[y] != 0) { y = t[y]; nivel++; }
    cout << nivel << ' ';
}
```

P10. Elevul caută frunzele ca „vârfuri de grad 1". Ce afișează pentru `t = (0, 1, 2, 2, 4)` ? Ce vârf e afișat greșit și de ce? Repară condiția.

```
for (int i = 1; i <= n; i++) if (t[i]) { g[i]++; g[t[i]]++; }
for (int i = 1; i <= n; i++) if (g[i] == 1) cout << i << ' ';
```

P11. *Puzzle cu rânduri amestecate.* Pune în ordine funcția de inserare într-un BST (duplicatele se ignoră). Un rând este în plus. Ce s-ar întâmpla dacă ai folosi rândul în plus în locul celui corect?

```
A)  else if (x > r->info) insereaza(r->dr, x);
B)  void insereaza(nod *&r, int x) {
C)      r = new nod; r->info = x; r->st = r->dr = NULL;
D)  }
E)  if (r == NULL) {
F)  void insereaza(nod *r, int x) {
G)  else if (x < r->info) insereaza(r->st, x);
H)  }
```

P12. Completează golurile: `unFiu` numără nodurile cu **exact un** fiu, iar `nrNivel(x, k)` numără nodurile de pe nivelul k din subarborele lui x. Ce dau `unFiu(1)` și `nrNivel(1, 2)` pentru arborele de la P5?

```
int unFiu(int x) {
    if (x == 0) return 0;
    int e = (st[x] == 0) != (__________);
    return e + unFiu(st[x]) + unFiu(dr[x]);
}
int nrNivel(int x, int k) {
    if (x == 0) return 0;
    if (__________) return 1;
    return nrNivel(st[x], ______) + nrNivel(dr[x], ______);
}
```

P13. *Ce greșeală a făcut elevul?* Dă câte un contraexemplu.

a. „Arborele este BST, pentru că la fiecare nod fiul stâng este mai mic și fiul drept mai mare."

b. „Din preordinea `1 2 3` și postordinea `3 2 1` reconstitui arborele în mod unic."

c. „Un arbore cu înălțimea 4 are 4 niveluri."

P14. Două funcții scrise de un elev pentru vectorii `st`, `dr`. Ce returnează pe arborele de la P5? Care sunt valorile corecte? Ce lipsește din fiecare?

```
int frunze(int x) {
    if (st[x] == 0 && dr[x] == 0) return 1;
    return frunze(st[x]) + frunze(dr[x]);
}
int inaltime(int x) {
    if (x == 0) return 0;
    return 1 + max(inaltime(st[x]), inaltime(dr[x]));
}
```

Nivelul 3 · Scrie cod

★★ scrie cod

La P15–P18 se citesc `n` și vectorul de tați. Verifică **toate** testele. Arborele **A**: n = 10, `t = (0, 1, 1, 1, 2, 2, 3, 4, 4, 6)`.

P15. Verii. Se citește și `x`. Afișează, crescător, verii primari ai lui x (au același bunic, dar alt tată), sau `niciunul`. Teste pe A: `5` → `7 8 9`; `8` → `5 6 7`; `10` → `niciunul`; `1` → `niciunul`.

P16. Organigrama. Afișează câți angajați sunt pe fiecare nivel (de la 0 la înălțime), apoi nivelul cu cei mai mulți (primul, la egalitate). Teste: A → `1 3 5 1` / `2`; T → `1 2 3 4` / `3`; `1` / `0` → `1` / `0`.

P17. Sistemul de fișiere. Se citesc și dimensiunile `s[1..n]` (fiecare folder are fișierele lui). Afișează folderul (nu rădăcina) cu cea mai mare dimensiune **totală** (cu tot subarborele lui) și dimensiunea. La egalitate, cel mai mic.

intrare	ieșire
A + `0 0 0 0 5 0 7 1 1 9`	`2 14`
`3` / `0 1 1` / `1 2 3`	`3 3`
`4` / `0 1 2 3` / `1 1 1 1`	`2 3`

P18. Validatorul. Afișează `DA` dacă vectorul este un vector de tați corect (exact un 0, celelalte valori între 1 și n, iar din orice vârf ajungi la rădăcină), altfel `NU`.

`5` / `0 1 1 2 2`	`5` / `0 1 0 2 2`	`4` / `2 3 2 0`	`3` / `0 1 4`	`1` / `0`	`4` / `2 0 4 3`
`DA`	`NU`	`NU`	`NU`	`DA`	`NU`

P19. Ghicitoarea. Se citesc `n` valori, care se inserează într-un BST, apoi `q`. Afișează înălțimea, apoi valorile din nodurile vizitate la căutarea lui `q`, apoi `gasit` sau `negasit`. Teste: valorile de la P6 și `33` → `3` / `40 25 30 35` / `negasit`; aceleași și `50` → `3` / `40 60 50` / `gasit`; `3` / `1 2 3` și `3` → `2` / `1 2 3` / `gasit`.

P20. Este BST? Se citește `n`, apoi pentru fiecare nod `v st dr`. Rădăcina **nu** este neapărat nodul 1: este nodul care nu e fiul nimănui. Afișează rădăcina și `DA` / `NU` (valorile sunt distincte).

intrare	ieșire
`4` / `10 2 3` / `5 0 4` / `15 0 0` / `12 0 0`	`1 NU`
`4` / `10 2 3` / `5 0 4` / `15 0 0` / `7 0 0`	`1 DA`
`3` / `20 0 0` / `10 3 1` / `5 0 0`	`2 DA`

Nivelul 4 · Pune mintea la contribuție

★★★ pune mintea la contribuție

P21. Problemă inversă. Scrie un vector de tați cu 7 noduri, înălțimea 3 și 4 frunze. Poate avea un asemenea arbore (7 noduri, înălțimea 3) 5 frunze? Care este numărul **minim** de frunze? Justifică.

P22. Sortarea cu arbore. Inserezi valorile într-un BST și le afișezi în inordine. De ce iese șirul sortat? Ce se întâmplă cu valorile care se repetă? Câte comparații face inserarea valorilor `1, 2, …, 10`, în această ordine, și ce înălțime are arborele? Ce ordine a inserării ar fi mult mai bună?

P23. Câte ordini dau același BST? Preordinea unui BST este `50 30 20 40 70 60 80`. Desenează-l (inordinea e chiar șirul sortat!) și scrie postordinea. Câte ordini de inserare a celor 7 valori produc **exact** acest arbore? (Indiciu: 50 trebuie să fie primul; apoi amesteci ordinea „stânga" cu ordinea „dreapta".)

P24. Strămoșul comun în BST. În BST-ul de la P6, cel mai apropiat strămoș comun al valorilor `a < b` este primul nod, coborând din rădăcină, cu `a ≤ info ≤ b`. De ce? Aplică pentru (5, 35), (30, 35), (10, 70), (50, 70) și (5, 10).

P25. Diametrul. Cel mai lung lanț dintre două vârfuri ale arborelui T are lungimea …? Care sunt capetele? Explică de ce, pentru fiecare vârf x, lanțul cel mai lung care are „vârful de sus" în x se obține din cei doi fii cu subarborii cei mai înalți.

Probleme de lucru în clasă (pbinfo)

Le rezolvăm împreună în ședință, în ordinea de mai jos.

- [ ] #636 **arbore** — ușoară
  https://www.pbinfo.ro/probleme/636/arbore
- [ ] #637 **frunze** — ușoară
  https://www.pbinfo.ro/probleme/637/frunze
- [ ] #639 **inaltime** — ușoară
  https://www.pbinfo.ro/probleme/639/inaltime
- [ ] #670 **preordine** — ușoară
  https://www.pbinfo.ro/probleme/670/preordine
- [ ] #671 **inordine** — ușoară
  https://www.pbinfo.ro/probleme/671/inordine

Temă pentru acasă (pbinfo)

Trimite soluțiile pe pbinfo și notează punctajul obținut. Bifează ce ai terminat.

- [ ] #638 **nivele** — ușoară
  https://www.pbinfo.ro/probleme/638/nivele
- [ ] #641 **subarbore** — ușoară
  https://www.pbinfo.ro/probleme/641/subarbore
- [ ] #644 **detdrum2** — ușoară
  https://www.pbinfo.ro/probleme/644/detdrum2
- [ ] #675 **bifrunze** — ușoară
  https://www.pbinfo.ro/probleme/675/bifrunze
- [ ] #3010 **bst** — medie
  https://www.pbinfo.ro/probleme/3010/bst