Arbori

BUZATU GIULIAN & NIȚĂ ALEXANDROS

Arbori

Un **arbore** este un graf conex și aciclic. Astfel, într-un arbore cu n noduri vom avea $n-1$ muchii. Dacă una din aceste muchii este eliminată, graful nu mai este conex, iar dacă mai este adăugată o muchie, se va obține un ciclu. Putem fixa un nod pe care să îl considerăm rădăcină, iar cu ajutorul acestui nod definim următoarele noțiuni:

- **Ascendent/Strămoș** al nodului u este orice nod v aflat pe lanțul ce leagă rădăcina de nodul u.
- **Tatăl** nodului u este ascendentul nodului pentru care există muchie între cele două.
  1. Rădăcina este singurul nod care este ascendent tuturor celorlalte și care nu are tată.
- **Descendent/Urmaș** al nodului u este orice nod v pentru care u este strămoș.
  1. Dacă există muchie între nodurile u și v de mai sus, atunci v este numit **fiul** lui u.
  2. Un nod care nu are fii se numește **frunză**.
- Definim **nivelul** lui u ca fiind lungimea unui lanț de la rădăcină la nodul u.
- Lungimea maximă a unui lanț de la rădăcină la un nod se numește **înălțimea** arborelui.
- Un nod al arborelui împreună cu toți descendenții săi formează un **subarbore**.
- Un arbore se numește **arbore binar**, dacă un nod are cel mult 2 fii.

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/913/B

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/913/B

Hint: Trebuie să calculăm numărul de frunze al fiecărui nod.

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/913/B

Rezolvare: Calculăm numărul de frunze pentru fiecare nod folosind un vector în care pentru fiecare nod u, $sons[u]$ = numărul de fii al nodului u. Dacă unul dintre acești fii are 0 fii, atunci, acesta este o frunză. Verificăm apoi pentru fiecare nod care nu este frunză dacă îndeplinește proprietatea din enunț, i. e. are mai mult de 3 fii.

Implementare: https://codeforces.com/contest/913/submission/249177923

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/862/B

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/862/B

Hint: Putem găsi o formulă pentru numărul de muchii pe care le putem adăuga?

Arbori

Să rezolvăm problema: https://codeforces.com/problemset/problem/862/B

Rezolvare: Vom împărți nodurile arborelui în două mulțimi ca în enunț. Aceasta se face printr-un DFS, alegând pentru fiecare nod mulțimea diferită de cea în care se află nodul tată al acestuia. Astfel obținem o bicolorare a nodurilor, formând graful bipartit.

Fie a și b cardinalele celor două mulțimi. Atunci numărul maxim de muchii ce pot exista între cele două mulțimi este ab, dar deja există $n-1$ muchii, deoarece graful este arbore. Atunci formula finală este $ab\ -n+1$.

Implementare: https://codeforces.com/contest/862/submission/249181527

Diametrul unui arbore

Să rezolvăm problema: https://infoarena.ro/problema/darb

Diametrul unui arbore

Să rezolvăm problema: https://infoarena.ro/problema/darb

Hint: Ce putem spune despre capetele diametrului?

Diametrul unui arbore

Să rezolvăm problema: https://infoarena.ro/problema/darb

Rezolvare: Cele două capete ale diametrului sunt frunze ale arborelui. Astfel, putem găsi un algoritm simplu de găsire a diametrului. Facem o parcurgere ca să aflăm cea mai îndepărtată frunză de rădăcină, după care mai facem o parcurgere pentru a găsi cea mai depărtată frunză de frunza găsită inițial.

Implementare: https://infoarena.ro/job_detail/3209149?action=view-source

Problemă

Să rezolvăm problema: https://codeforces.com/contest/1881/problem/F

Problemă

Să rezolvăm problema: https://codeforces.com/contest/1881/problem/F

Hint: Care este răspunsul pentru un arbore cu doar două noduri roșii?

Problemă

Să rezolvăm problema: https://codeforces.com/contest/1881/problem/F

Rezolvare: Dacă am avea doar două noduri roșii, putem observa că răspunsul se va afla la mijlocul lanțului care le unește. Putem intui că, pentru mai multe noduri roșii, răspunsul se va afla la mijloului „diametrului” format din nodurile roșii. Adică, pe lanțul de lungime maximă dintre două noduri roșii.

Implementare: https://codeforces.com/contest/1881/submission/230843081

Liniarizări pe arbore

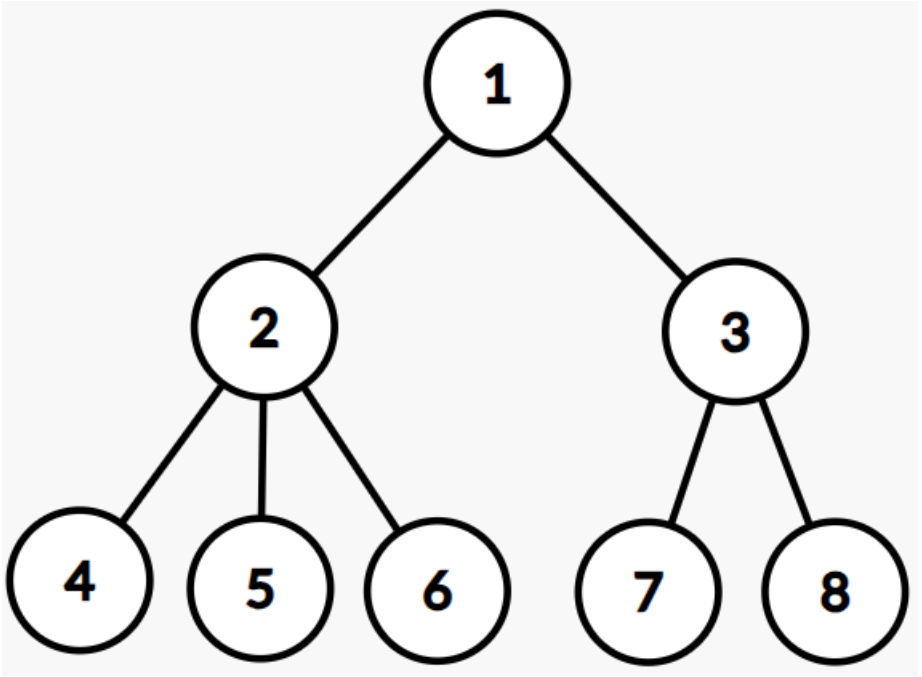
Uneori avem nevoie să lucrăm cu arborii într-o formă liniară. Astfel, apare conceptul de liniarizare, prin care ne propunem să ținem arborele într-o anumită formă pentru a putea lucra mai ușor cu anumite informații pe care le ținem în noduri sau pe care le știm despre noduri. Există mai multe feluri de liniarizare, care merg pe orice fel de arbori, printre care următoarele:

1. Preordine
2. Postordine
3. Euler
4. 0-1
5. Magică

Observație: Pentru arborii binari există și liniarizarea înordine.

Preordine

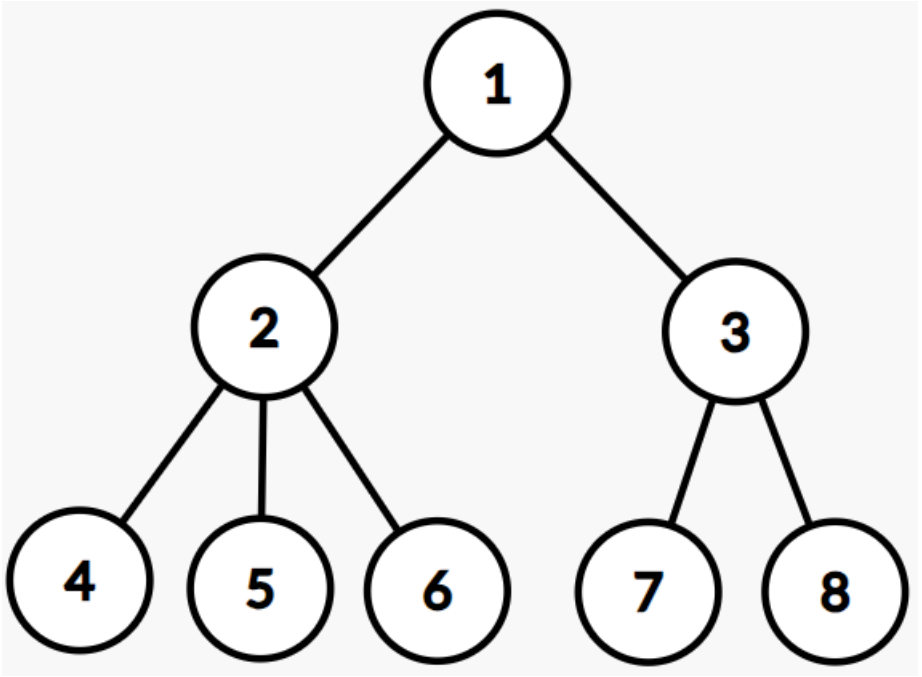
Punem în vector rădăcina, după care parcurgem primul subarbore, după care al doilea, etc.
Pentru arborele alăturat, parcurgerea arată astfel: 1 2 4 5 6 3 7 8.



Problemă: https://pbinfo.ro/probleme/670
Implementare: https://github.com/Giulian617/Hai-la-olimpiada-2023-2024/blob/main/11-12/resources/preordine.cpp
Observație: Fiecare nod apare o singură dată.

Postordine

Punem nodurile din primul subarbore, după care al doilea, etc. După aceea punem rădăcina subarborelui curent. Pentru arborele alăturat, parcurgerea arată astfel: 4 5 6 2 7 8 3 1.

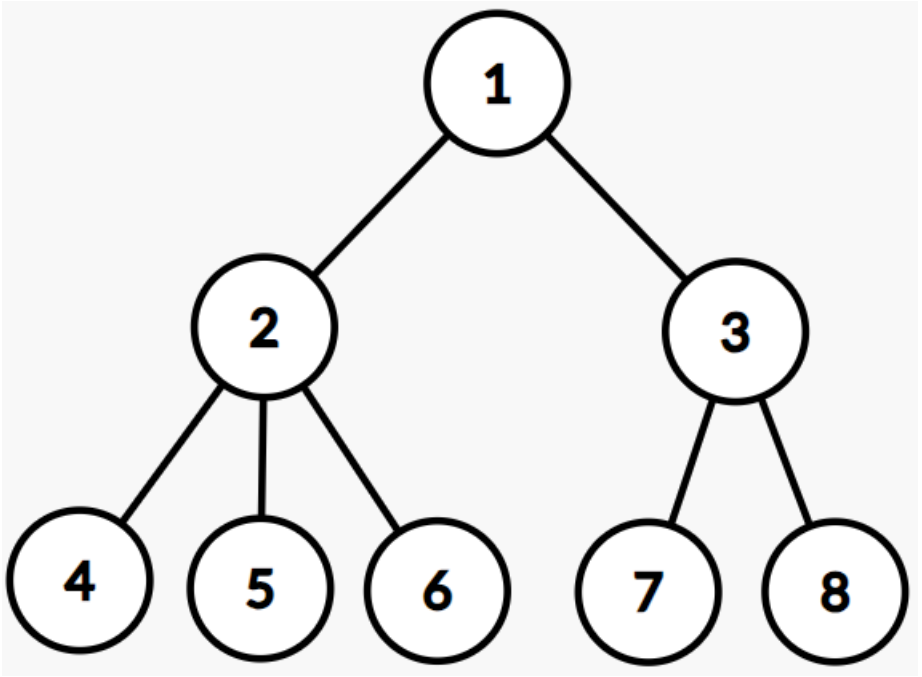


Problemă: https://pbinfo.ro/probleme/672
Implementare: https://github.com/Giulian617/Hai-la-olimpiada-2023-2024/blob/main/11-12/resources/postordine.cpp
Observație: Fiecare nod apare o singură dată.

Euler

Atunci când facem DFS, scriem un nod de fiecare dată când trecem prin el. Pentru arborele alăturat, parcurgerea arată astfel: 1 2 4 2 5 2 6 2 1 3 7 3 8 3 1.

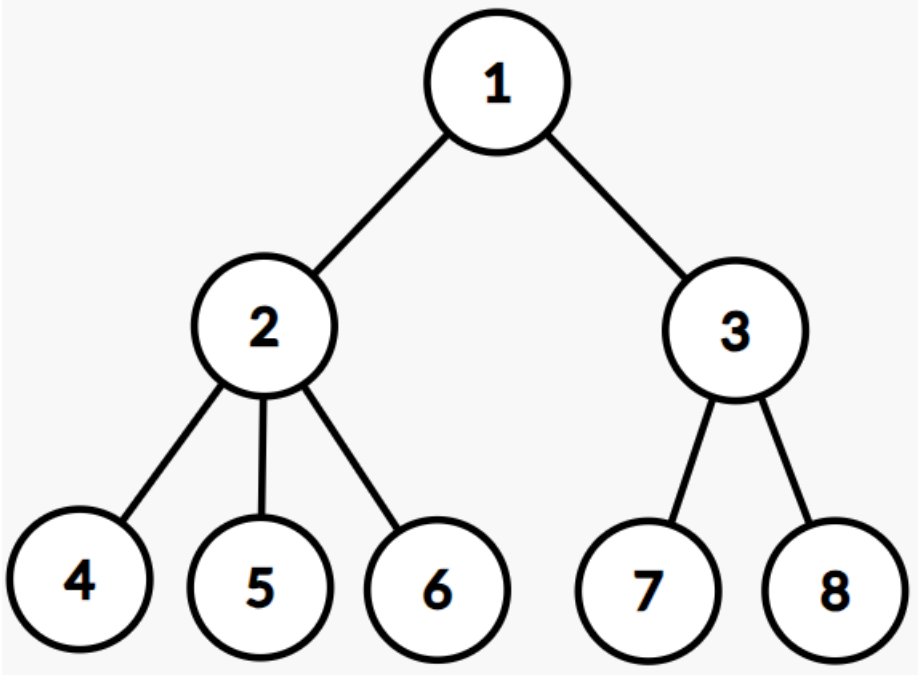
Observație 1: Un nod apare de mai multe ori,anume de nr. fii+1 ori.
Observație 2: Se folosește la LCA.



0-1

Facem DFS, iar când intrăm în nod scriem 1 și când ieșim scriem 0. Pentru arborele alăturat, parcurgerea arată astfel (indicele indică nodul care scrie 1 sau 0):
$1_1 1_2 1_4 0_4 1_5 0_5 1_6 0_6 0_2 1_3 1_7 0_7 1_8 0_8 0_3 0_1$.

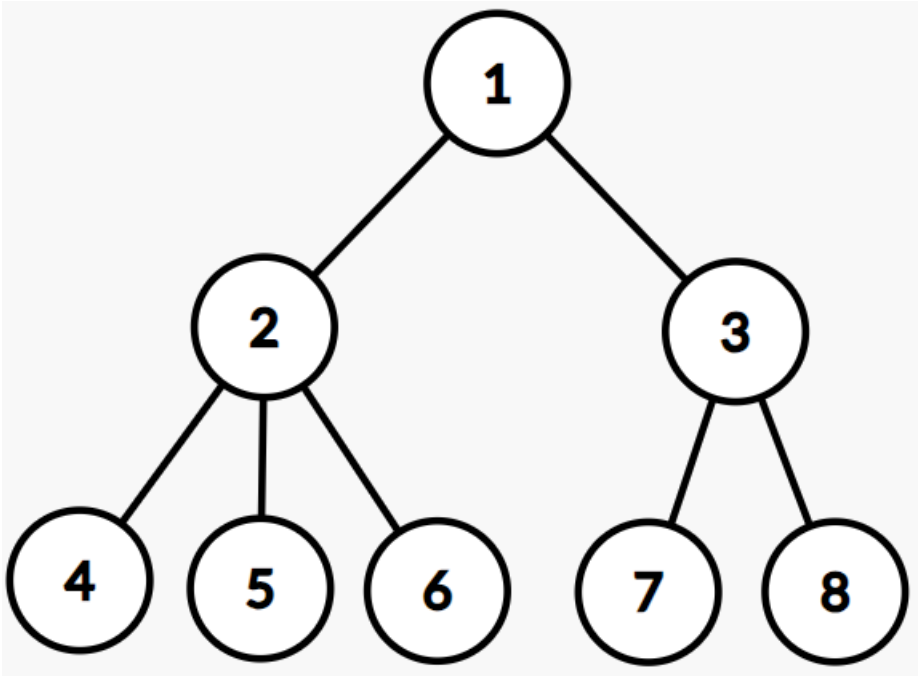
Observație 1: Poate fi privită ca o parantezare, deoarece fiecare nod scrie 1 exact o dată și 0 exact o dată.
Observație 2: Se folosește la izomorfism de arbori.



Magică

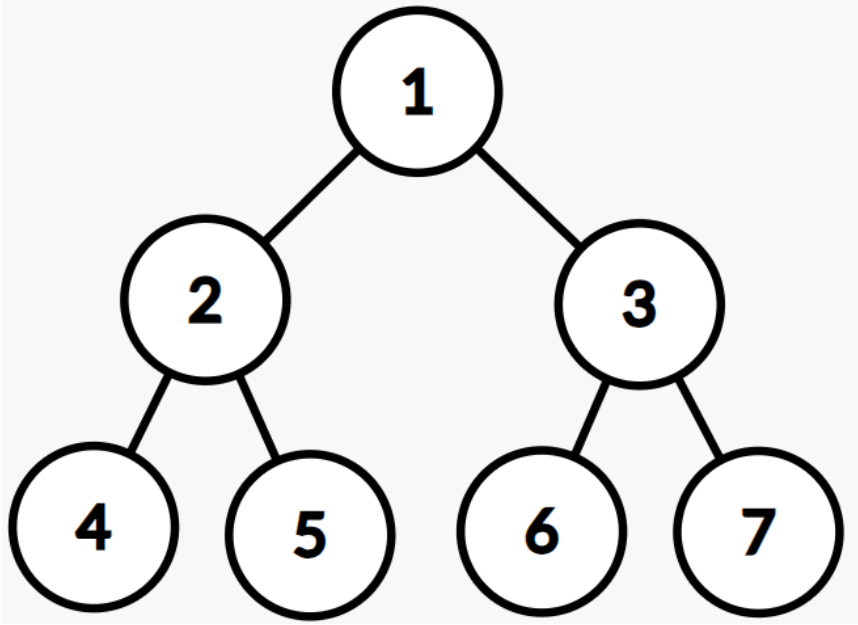
Facem DFS, iar când intrăm în nod îl scriem și procedăm la fel la ieșire. Pentru arborele alăturat, parcurgerea arată astfel: 1 2 4 4 5 5 6 6 2 3 7 7 8 8 3 1.

Observație: Fiecare nod apare de exact două ori.



Înordine

Parcurgem subarborele stâng, după care scriem rădăcina și, după, parcurgem subarborele drept.
Pentru arborele alăturat, parcurgerea arată astfel: 4 2 5 1 6 3 7.
Problemă: https://pbinfo.ro/probleme/671
Implementare: https://github.com/Giulian617/Hai-la-olimpiada-2023-2024/blob/main/11-12/resources/inordine.cpp
Observație 1: Fiecare nod apare o singură dată.
Observație 2: Arborele trebuie să fie binar.



DP pe arbori

Să rezolvăm problema: https://infoarena.ro/problema/asmax

DP pe arbori

Să rezolvăm problema: https://infoarena.ro/problema/asmax

Hint: Ce avem nevoie pentru calcularea sumei unui subarbore ce pornește din nodul u?

DP pe arbori

Să rezolvăm problema: https://infoarena.ro/problema/asmax

Rezolvare: Suma unui subarbore ce începe din nodul u se obține adunând sumele obținute din subarborii fiilor nodului u. Dintre aceste sume trebuie, însă, să le păstrăm doar pe cele pozitive, pentru a maximiza suma totală. Vom folosi un DFS pentru parcurgerea arborelui și obținerea acestor sume.

Implementare: https://infoarena.ro/job_detail/3209021?action=view-source

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/easygraph

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/easygraph

Hint: Dacă suntem în nodul u, ce drum ne aduce suma maximă?

Problemă

Să rezolvăm problema: https://infoarena.ro/problema/easygraph

Rezolvare: Fiind în nodul u, suma maximă pe care o putem calcula se obține alegând drumul cu suma maximă începând cu nodurile adiacente ale lui u. Evident, alegem suma maximă cu proprietatea că e pozitivă, pentru a maximiza suma. Parcurgem graful cu DFS pentru a calcula aceste sume.

Implementare: https://infoarena.ro/job_detail/3209029?action=view-source

Observație: Aceasta este o problemă cu DAG (directed acyclic graph), dar abordările folosite sunt foarte asemănătoare cu cele de la arbori.

Temă

- https://infoarena.ro/problema/euler
- https://codeforces.com/problemset/problem/580/C
- https://codeforces.com/problemset/problem/1143/C
- https://codeforces.com/problemset/problem/982/C

Probleme suplimentare

- https://infoarena.ro/problema/tairos
- https://codeforces.com/problemset/problem/734/E

Lectură suplimentară

- https://usaco.guide/gold/dp-trees
- https://usaco.guide/gold/all-roots