Problema Keidei

Fişier de intrare	`keidei.in`
Fişier de ieşire	`keidei.out`

Ranuto are un program de antrenament de-a lungul a mai multor zile, în care vrea să învețe o nouă tehnică: "11th Gate Super Convincing Chakra Dual-Core Rasengan". Din motive obiective, vrea să știe din avans în ce ar putea consta antrenamentul lui într-o zi anume (vine Kasura să se uite la el).

Se dă un arbore cu N noduri, numerotate de la 1 la N. Arborele este înrădăcinat în nodul 1.

Vrem să facem o parcurgere a arborelui, pornind din rădăcină. Pentru fiecare nod, putem considera fiii acestuia în orice ordine dorim.

Există două tipuri de cerințe, reprezentate printr-un număr c:

- Pentru $C = 1$, parcurgerea va fi de tip **adâncime (DFS) pre-ordine**.
- Pentru $C = 2$, parcurgerea arborelui va fi de tip **lățime (BFS)**.

Pseudocodul pentru parcurgeri poate fi consultat mai jos:

Algoritmul 1 Parcurgere în adâncime (DFS) pre-ordine

```
p ← listă goală (va conține nodurile din parcurgere)
procedură DFS(arbore, nod):
    adaugă nod în spatele lui p
    pentru fiecare fiu x al lui nod:
        DFS(arbore, x)
DFS(arbore, 1)
```

Algoritmul 2 Parcurgere în lățime (BFS)

```
p ← listă goală (va conține nodurile din parcurgere)
procedură BFS(arbore):
    q ← coadă goală
    adaugă nodul 1 în spatele lui q
    cât timp q nu este goală:
        f ← nodul din fața lui q
        scoate f din fața lui q
        adaugă f în spatele lui p
        pentru fiecare fiu x al lui f:
            adaugă nodul x în spatele lui q
BFS(arbore)
```

Care noduri din arbore pot să fie pe a K-a poziție în vreuna dintre posibilele parcurgeri?

Date Intrare

Pe prima linie se găsesc trei numere, C, N și K. Pe următoarele $N - 1$ linii se găsesc câte 2 numere, A și B, semnificând existența unei muchii în arbore între nodurile A și B.

Date Ieșire

Afișaţi **pe o linie, cu un spaţiu între ele, în ordine crescătoare**, toate nodurile care pot fi pe a K-a poziție în vreo parcurgere de tipul stabilit de C. **Nu trebuie afișat numărul de noduri.**

Restricții

- $1 \leq N \leq 10\,000$.
- $1 \leq K \leq N$.

#	Punctaj	Restricţii
1	5	$C = 1$, $N \leq 10$
2	5	$C = 2$, $N \leq 10$
3	5	$C = 1$. Arborele este binar complet, cu N de forma $2^p - 1$, p număr natural nenul.
4	5	$C = 2$. Arborele este binar complet, cu N de forma $2^p - 1$, p număr natural nenul.
5	20	$C = 1$, $N \leq 500$
6	20	$C = 2$, $N \leq 500$
7	15	$C = 1$, $N \leq 3\,500$
8	10	$C = 2$, $N \leq 5\,000$
9	15	$C = 1$, $N \leq 10\,000$

Exemplul 1		Exemplul 2		Exemplul 3	
keidei.in	**keidei.out**	**keidei.in**	**keidei.out**	**keidei.in**	**keidei.out**
1 8 5 1 2 1 3 1 4 2 5 2 6 3 7 5 8	2 5 6 8	1 17 15 1 2 1 3 1 4 2 5 3 6 3 7 4 8 5 9 5 10 8 11 8 12 8 13 8 14 9 15 10 16 10 17	3 10 11 12 13 14 16 17	2 8 5 1 2 1 3 2 4 2 5 2 7 2 8 3 6	4 5 7 8

Explicații

Pentru primul exemplu: Pentru că putem să considerăm fiii oricărui nod din arbore în ce ordine dorim, putem să presupunem că parcurgerea noastră în adâncime parcurge fiii unui nod de la stânga la dreapta după ce stabilim ordinea.

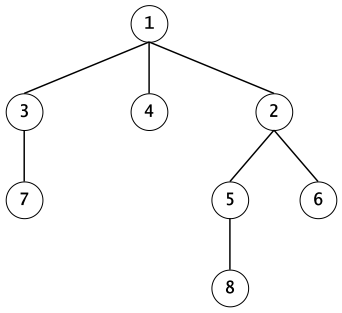

Figura 1: Dacă rearanjăm arborele primit ca să arate ca in figura de mai sus, parcurgerea va produce următorul rezultat: $1, 3, 7, 4, \mathbf{2}, 5, 8, 6$.

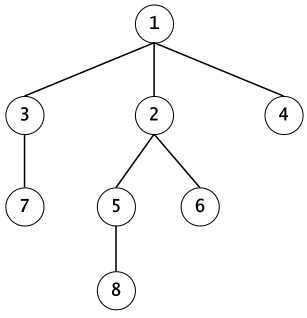

Figura 2: Aici, parcurgerea va fi $1, 3, 7, 2, \mathbf{5}, 8, 6, 4$.

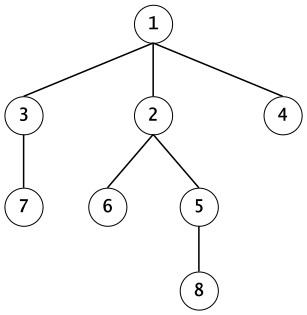

Figura 3: Aici, parcurgerea va fi $1, 3, 7, 2, \mathbf{6}, 5, 8, 4$.

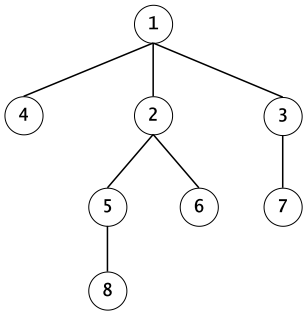

Figura 4: Aici, parcurgerea va fi $1, 4, 2, 5, \mathbf{8}, 6, 3, 7$.

Putem verifica orice alt aranjament pentru a garanta că orice alt nod în afară de $2, 5, 6, 8$ nu poate fi pe a 5-a poziție într-o parcurgere în adâncime pre-ordine.

Pentru al treilea exemplu, putem observa că oricum am aranja arborele, primele 3 noduri din parcurgere vor fi ori $1, 2, 3$, ori $1, 3, 2$. Dacă 2 este înaintea lui 3 în parcurgere, atunci 6 va fi neapărat ultimul nod din parcurgere. Dacă 3 ar fi înaintea lui 2, atunci 6 va fi neapărat pe a patra poziție în parcurgere. Astfel, singurele noduri care pot fi pe a 5-a poziție sunt $4, 5, 7, 8$.