



PROBLEMA ENGINEERS

Este timpul pentru mentenanța anuală a serverelor la *Vianu*. O parte a lucrărilor de mentenanță pot fi realizate de oameni, altele nu. De exemplu, cablurile rețelei sunt trasate subteran prin tuburi de diametru mic inaccesibil oamenilor! Din fericire, un inginer experimentat a venit cu ideea de a folosi șobolani dresați pentru verificarea calculatoarelor.

Fiecare șobolan intră în rețea de la un calculator, se mișcă de-a lungul cablurilor prin unul sau mai multe tuburi, verificând toate calculatoarele întâlnite în calea sa și iese din rețea la alt calculator. Deoarece șobolanii sunt destul de leneși, fiecare dintre ei va urma o singură cale simplă între punctele de intrare și ieșire, înainte de a obosi.

Echipa de mentenanță dorește ca șobolanii să verifice unele calculatoare altfel încât configurațiile codurilor de securitate a calculatoarelor rămase neverificate să fie “suficient de balansate”. Sarcina ta este de a determina numărul minim de șobolani necesari echipei.

■ **PROBLEMĂ** Sunt N calculatoare, numerotate de la 0 to $N - 1$, legate prin $N - 1$ cabluri. Rețeaua este un arbore (conectat și aciclic).

Un șobolan verifică fiecare calculator de-a lungul drumului simplu unic dintre calculatorul de intrare în rețea și cel de la ieșire din rețea. Astfel, dacă un șobolan intră în rețea prin calculatorul S și iese la calculatorul T ($S, T \in \{0, \dots, N - 1\}$), atunci șobolanul va verifica toate calculatoarele $v_1, v_2, \dots, v_k$ astfel încât:

- $v_1 = S$ și $v_k = T$,
- Pentru toate calculatoarele $1 \leq i < k$, v_i și v_{i+1} sunt conectate direct prin cablu,
- k este minim.

Șobolanii pot refolosi aceleași cabluri și calculatoare, astfel un calculator poate fi verificat de mai mulți șobolani.

Fiecare calculator $i \in \{0, \dots, N - 1\}$ are asociat un cod întreg pozitiv $C_i \geq 1$. Echipa de mentenanță fixează o diferență maximal acceptabilă D . Fie, după ce toți șobolanii își finalizează lucrul, setul de calculatoare rămase (neverificate) va fi $R \subseteq \{0, \dots, N - 1\}$. Echipa adorește să se asigure că $|C_i - C_j| \leq D$ pentru orice pereche $i, j \in R$.

Cu alte cuvinte, diferența dintre codul maximal și codul minimal a calculatoarelor rămase trebuie să nu depășească D .

Fiecare șobolan va verifica o singură cale înainte să obosească. Va trebui să aflați numărul minim de șobolani folosiți.

■ **IMPLEMENTARE** Va trebui să implementezi funcția:

```
int solve(int N, int D, std::vector<int> C, std::vector<int> P, std::vector<int> Q)
```

Această funcție primește ca parametri: N , numărul de calculatoare, D , diferența maximală acceptată, C , codurile calculatoarelor, și doi vectori de lungime $N - 1$, P and Q , însemnând că există un cablu între calculatoarele P_i și Q_i , pentru toate valorile i , unde $0 \leq i \leq N - 1$.



Funcția va returna numărul minim de șobolani necesari, astfel încât după verificarea rețelei de către aceștia, calculatoarele neverificate respectă relația:

$$\max_{i \in R} C_i - \min_{i \in R} C_i \leq D.$$

- **RESTRICȚII**
- ◆ $1 \leq N \leq 200\,000$.
 - ◆ $1 \leq C_i \leq 1\,000\,000\,000$ pentru toate $0 \leq i \leq N - 1$.
 - ◆ $1 \leq D \leq 1\,000\,000\,000$.
 - ◆ $0 \leq p_i, q_i < N$, $p_i \neq q_i$, și toate perechile (p_i, q_i) sunt distincte.

■ **SUBTASKURI**

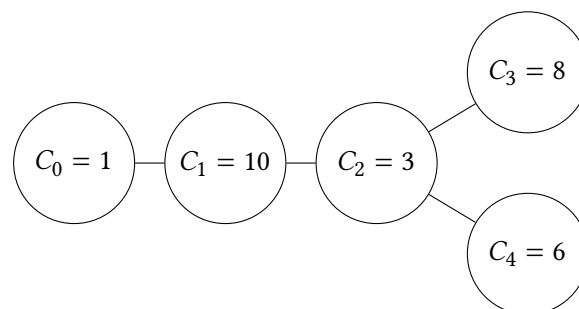
#	Punctaj	Restricții
1	7	$N \leq 20$ și $1 \leq C_i \leq 50$ pentru toți $0 \leq i \leq N - 1$.
2	6	$N \leq 1000$ și $1 \leq C_i \leq 1000$ pentru toți $0 \leq i \leq N - 1$.
3	11	$N \leq 1000$.
4	16	$p_i = 0, q_i = i + 1$ pentru toți $0 \leq i < N - 1$.
5	26	$N \leq 50000$.
6	34	Fără restricții adiționale



EXAMPLE

Date de intrare	Date de ieșire
5 3	1
1 10 3 8 6	
0 1	
1 2	
2 3	
2 4	
20 30	3
13 36 11 35 4 9 42 9 1 4 11 3 15 31	
→ 46 41 31 17 11 12	
19 5	
19 0	
19 13	
19 9	
19 4	
19 10	
5 1	
19 18	
0 7	
5 8	
19 12	
5 17	
13 16	
5 14	
13 3	
19 6	
5 15	
5 2	
4 11	

În primul exemplu sunt $N = 5$ calculatoare și $D = 3$. Rețeaua poate fi vizualizată astfel:



O strategie posibilă este de a trimite un singur șobolan care să-și înceapă drumul la calculatorul 0 și să îl termine la calculatorul 3, trecând pe la calculatoarele 0 – 1 – 2 – 3. După o asemenea verificare, rețeaua este sigură.