Stations (stations)

Singapore's Internet Backbone (SIB) constă în n stații, cărora li s-au atribuit **indici** între 0 și $n-1$. De asemenea, există $n-1$ legături bidirecționale, numerotate de la 0 la $n-2$. Fiecare legătură conectează două stații distincte. Două stații conectate printr-o legătură se numesc vecine.

Un drum de la stația x la stația y este o secvență de stații distincte $a_0, a_1, \cdots, a_p$, astfel încât $a_0 = x$, $a_p = y$ și oricare două stații consecutive pe drum sunt vecine. Există **exact un** drum între oricare doua stații distincte x și y.

Orice stație x poate crea un pachet (de date) și să îl trimită către oricare altă stație y, numită **destinația** pachetului. Acest pachet trebuie rutat pe unicul drum de la x la y, după cum urmează. Fie z stația ce deține la momentul actual pachetul, a cărui stație destinație este y ($z \neq y$). În acest caz stația z:

1. Execută o **procedură de rutare** ce determină vecinul lui z care se află pe unicul drum de la z la y, și
2. trimite mai departe pachetul către acest vecin.

Cu toate acestea, stațiile au la dispoziție o cantitate limitată de memorie și nu pot reține întreaga listă de legături din SIB pentru a o folosi la procedura de rutare.

Sarcina voastră este să implementați o schemă de rutare pentru SIB, care constă în două proceduri:

- Prima procedură primește n, lista de legaturi din SIB și un întreg $k \geq n - 1$. Aceasta atribuie fiecărei stații un întreg **unic**, numit **eticheta** (eng. **label**), între 0 și k, inclusiv.
- A doua procedură este procedura de rutare, care este lansată către toate stațiile după ce etichetele au fost atribuite. Aceasta primește **doar** următoarele intrări:
    - s, **eticheta** stației ce deține pachetul la momentul actual,
    - t, **eticheta** stației destinație a pachetului ($t \neq s$),
    - c, lista **etichetelor** tuturor vecinilor lui s.

  Aceasta va returna **eticheta** acelui vecin al lui s căruia îi trebuie trimis mai departe pachetul.

Într-un subtask, scorul submisiei dvs. depinde de eticheta maximă atribuită unei stații (în general, cu cât mai mic, cu atât mai bine).

Detalii de implementare

Trebuie să implementați următoarele proceduri:

```
int[] label(int n, int k, int[] u, int[] v)
```

- n: numărul de stații din SIB.
- k: valoarea maximă a unei etichete ce poate fi folosită.
- u și v: tablouri de mărime $n-1$ ce descriu legăturile din SIB. Oricare ar fi i ($0 \leq i \leq n-2$), legătura i conectează stațiile cu indicii $u[i]$ și $v[i]$.
- Procedura va returna un singur tablou L de lungime n. Oricare ar fi i ($0 \leq i \leq n-1$) $L[i]$ va reprezenta eticheta atribuită statiei de indice i. Elementele tabloului L trebuie sa fie unice si cuprinse intre 0 si k, inclusiv.

```
int find_next_station(int s, int t, int[] c)
```

- s: eticheta stației ce deține pachetul.
- t: eticheta stației destinație a pachetului.
- c: un tablou reprezentând etichetele vecinilor lui s. Tabloul c va fi sortat crescător.
- Procedura va returna eticheta vecinului lui s căruia îi trebuie trimis mai departe pachetul.

Fiecare test constă în unul sau mai multe scenarii independente (adică diferite descrieri ale SIB). Pentru fiecare test constant în r scenarii, un **program** ce apelează procedurile de mai sus este rulat de exact două ori, după cum urmează.

La prima rulare a programului:

- procedura `label` este apelată de r ori,
- etichetele returnate sunt reținute de sistemul de evaluare, și
- `find_next_station` nu este apelată.

La a doua rulare a programului:

- `find_next_station` poate fi apelata de mai multe ori. La fiecare apel este ales un scenariu **arbitrar**, iar apoi etichetele returnate de procedura `label` pentru acel scenariu sunt folosite ca intrări pentru `find_next_station`.
- `label` nu este apelata.

Orice informație reținută în variabile statice sau globale la prima rulare nu va fi disponibila în procedura `find_next_station`.

Exemplu

Considerăm următorul apel:

```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

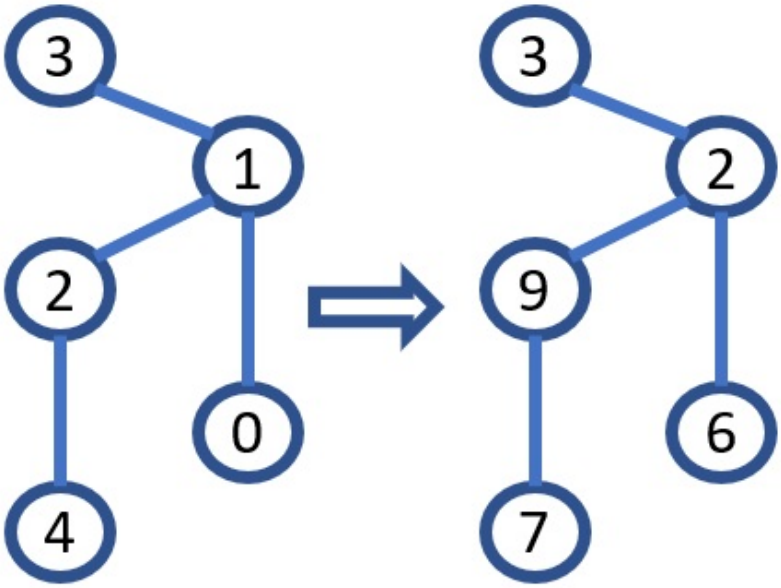
În total sunt 5 stații și 4 legături conectând perechile de stații cu indicii $(0, 1)$, $(1, 2)$, $(1, 3)$ și $(2, 4)$.

Etichetele pot fi orice număr întreg între 0 și $k = 10$.

Pentru a raporta următoarea etichetare:

Indice	Eticheta
0	6
1	2
2	9
3	3
4	7

procedura `label` trebuie să returneze [6, 2, 9, 3, 7]. Numerele din următoarea figură arată indicii (în stânga) și etichetele atribuite (în dreapta).



Să presupunem că etichetele au fost atribuite ca mai sus și să considerăm următorul apel:

```
find_next_station(9, 6, [2, 7])
```

Acesta înseamnă ca stația ce deține pachetul are eticheta 9, iar stația destinație a pachetului are eticheta 6. Etichetele stațiilor de pe drumul până la stația destinație sunt $[9, 2, 6]$. Prin urmare, apelul trebuie să returneze 2, reprezentând eticheta stației către care pachetul trebuie trimis mai departe (aceasta având indice 1).

Să considerăm încă un posibil apel:

```
find_next_station(2, 3, [3, 6, 9])
```

Procedura trebuie să returneze 3, deoarece stația destinație, având eticheta 3, este vecin al stației cu eticheta 2, și prin urmare poate primi pachetul direct.

Restricții

- $1 \leq r \leq 10$

Pentru fiecare apel al procedurii `label`:

- $2 \leq n \leq 1000$
- $k \geq n - 1$
- $0 \leq u[i], v[i] \leq n - 1$ (oricare ar fi $0 \leq i \leq n - 2$)

Pentru fiecare apel al procedurii `find_next_station`, intrarea provine de la un apel anterior arbitrar ales al procedurii `label`. Să considerăm etichetele produse. Atunci:

- s și t sunt etichetele a două stații diferite.
- c reprezintă secvența de etichete a vecinilor stației cu eticheta s, în ordine crescătoare.

Pentru fiecare test, lungimea totala a tuturor tablourilor c primite de procedura `find_next_station` nu va depăși $100\ 000$ pentru toate scenariile la un loc.

Subtaskuri

1. (5 puncte) $k = 1000$, nicio stație nu va avea mai mult de 2 vecini.
2. (8 puncte) $k = 1000$, legatura i conectează stațiile $i + 1$ și $\left\lfloor \frac{i}{2} \right\rfloor$.
3. (16 puncte) $k = 1\ 000\ 000$, cel mult o stație va avea mai mult de 2 vecini.
4. (10 puncte) $n \leq 8$, $k = 10^9$
5. (61 de puncte) $k = 10^9$

În subtaskul 5 puteți obține punctaj parțial. Fie m eticheta maximă returnată de procedura `label`, luând în calcul toate scenariile. Scorul dvs. pentru acest subtask este calculat conform cu următorul tabel:

Etichetă Maximă	Scor
$m \geq 10^9$	0
$2000 \leq m < 10^9$	$50 \cdot \log_{5 \cdot 10^5}(\frac{10^9}{m})$
$1000 < m < 2000$	50
$m \leq 1000$	61

Sample grader

Sample graderul citește intrarea în următorul format:

- linia 1: r

r blocuri urmează, fiecare descriind câte un singur scenariu. Formatul fiecărui bloc este după cum urmează:

- linia 1: n k
- linia $2 + i$ $(0 \leq i \leq n - 2)$: $u[i]$ $v[i]$
- linia $1 + n$: q: numărul de apeluri ale procedurii `find_next_station`.
- linia $2 + n + j$ $(0 \leq j \leq q - 1)$: $z[j]$ $y[j]$ $w[j]$: **indicii** stațiilor implicate în al j-ulea apel al procedurii `find_next_station`. Stația $z[j]$ deține pachetul, stația $y[j]$ este destinația pachetului, iar stația $w[j]$ este accea către care pachetul trebuie trimis mai departe.

Sample graderul afișează rezultatul în următorul format:

- linia 1: m

r blocuri ce corespund la scenarii consecutive din input. Formatul fiecărui bloc este după cum urmează:

- linia $1 + j$ $(0 \leq j \leq q - 1)$: **indicele** stației a cărei **etichetă** a fost returnată de către al j-ulea apel al `find_next_station` in acest scenariu.

Băgați de seamă cum fiecare rulare a sample graderului apelează atat `label` cat si `find_next_station`.