

Serverové stanice (stations)

Kostra internetu v Singapure je tvorená n servermi, ktoré voláme stanice. Tie majú čísla od 0 po $n - 1$. Medzi stanicami vedie $n - 1$ obojsmerných liniek. Linky sú vedené tak, že je celá sieť súvislá, a teda má stromovú topológiu. Ak sú dve stanice prepojené linkou, hovoríme, že susedia.

Hocijaká stanica x môže poslať paket, ktorého **adresátom** môže byť ľubovoľná iná stanica y . Každý paket **musí** byť poslaný po unikátnej ceste vedúcej z x do y (nesmie ísť dvakrát tou istou hranou). Formálne, cesta je postupnosť **navzájom rôznych** staníc a_0, a_1, \dots, a_p taká, že $a_0 = x$, $a_p = y$, a každé dve po sebe idúce stanice (a_i, a_{i+1}) susedia. (Číslo p udáva počet liniek na tejto ceste.)

Existuje jedna **rútovacia funkcia**; program, ktorý beží na každej stanici. Keď máme paket, ktorý je na ceste sieťou v stanici a_i ($0 \leq i < p$) jeho cesty, rútovacia funkcia vypočíta **nasledujúcu stanicu** a_{i+1} , na ktorú má paket pokračovať. Potom stanica a_i prepošle príslušnou linkou tento paket na stanicu a_{i+1} .

Takúto rútovaciu funkciu by sme ľahko vedeli vyrobiť za predpokladu, že by mala prístup k popisu celého singapurského internetu (teda počtu staníc a zoznamu liniek). My to ale chceme spraviť šikovnejšie. Každéj stanici priradíme nejaký číselný **identifikátor**, a to tak, aby sme potom vedeli rútovať pakety lokálne.

Formálne, vašou úlohou bude naprogramovať dve funkcie:

- **Značkovacia funkcia** dostane na vstupe popis celého internetu a tiež celé číslo $k \geq n - 1$. Táto funkcia musí každej stanici priradiť jej identifikátor: celé číslo z rozsahu od 0 po k , vrátane. Všetky identifikátory musia byť **navzájom rôzne**.
- **Rútovacia funkcia** bude zavolaná vždy, keď máme paket v stanici, na ktorú ešte nepatrí. Táto funkcia dostane **len** nasledujúce vstupy:
 - s : **identifikátor** stanice, v ktorej sa paket nachádza
 - t : **identifikátor** stanice, ktorá je koncovým adresátom tohto paketu
 - c : zoznam **identifikátorov** všetkých susedov aktuálnej stanice s Rútovacia funkcia by mala vrátiť identifikátor tej susednej stanice, na ktorú treba tento paket preposlať.

Upozorňujeme, že obe funkcie sú úplne samostatné. Špecificky, rútovacia funkcia nemá prístup k ničomu inému okrem svojich parametrov: ani k popisu celého internetu, ani ku hodnotám premenných, ktoré používala vaša značkovacia funkcia.

Váš výsledný počet bodov bude závisieť od toho, pre akú hodnotu maximálneho identifikátora k úlohu zvládnete riešiť.

Detaily implementácie

Naprogramujte nasledujúce dve funkcie:

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

- n : celkový počet staníc tvoriacich singapurský internet
- k : maximálna povolená hodnota identifikátora
- u a v : polia dĺžky $n - 1$ popisujúce jednotlivé linky. Pre každé i ($0 \leq i \leq n - 2$) platí, že linka i spája stanice $u[i]$ a $v[i]$.
- Návratovou hodnotou musí byť pole dĺžky n . Na indexe i v tomto poli má byť identifikátor, ktorý chcete priradiť vrcholu číslo i . Všetky prvky tohto poľa teda musia byť navzájom rôzne a musia byť z rozsahu od 0 po k , vrátane.

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

- s : identifikátor stanice, kde sa paket nachádza
- t : identifikátor stanice, na ktorej chce paket svoju cestu ukončiť
- c : pole dĺžky l obsahujúce identifikátory všetkých susedov stanice s . Toto pole bude vždy usporiadané v rastúcom poradí.
- Na výstupe má táto funkcia vrátiť identifikátor toho suseda stanice s , ktorý leží na jedinej ceste zo stanice s identifikátorom s do stanice s identifikátorom t .
- Túto funkciu bude grader volať veľa krát (viď nižšie). Je zaručené, že súčet dĺžok všetkých polí c pri týchto volaniach neprekročí 100 000.

Každý test bude obsahovať niekoľko nezávislých scenárov. Ich počet označme r . Počas vyhodnotenia testu budú postupne spustené dva programy.

Prvý spustený program:

- postupne r -krát zavolá vašu funkciu `label`
- pre každé volanie si uloží pole s identifikátormi, ktoré vaša funkcia vyrobila
- tento program vôbec nevolá funkciu `find_next_station`

Druhý spustený program

- bude niekoľkokrát volať funkciu `find_next_station`
- pri každom volaní `find_next_station` budú identifikátory, ktoré dostanete ako parametre, zodpovedať jednému **ľubovoľne zvolenému scenáru** spomedzi tých, ktoré vaša funkcia `label` spracovala počas prvého behu programu.
- tento program vôbec nevolá funkciu `label`

Ukázkový príklad

Prvý program zavolá vašu funkciu `label` nasledovne:

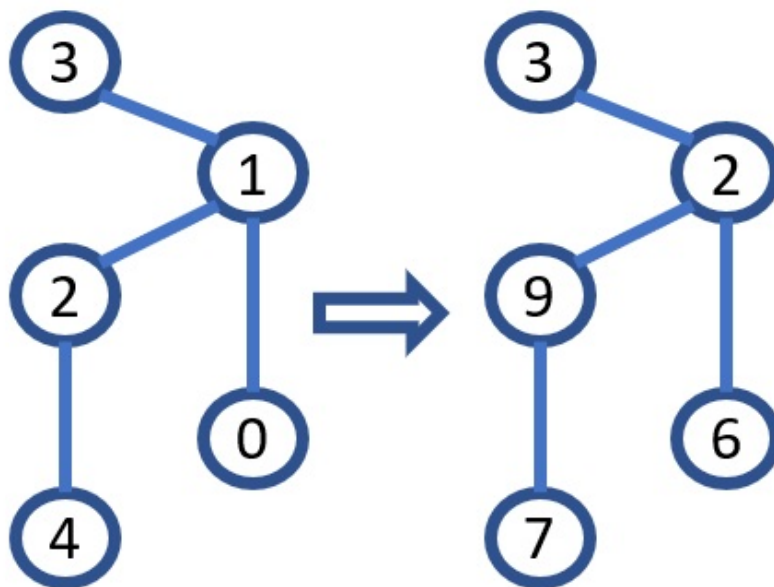
```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

Existuje dokopy 5 staníc, sú prepojené štyrmi linkami. Dvojice susedných staníc sú (0, 1), (1, 2), (1, 3) a (2, 4). Ako identifikátory sme si mohli použiť čísla od 0 po $k = 10$.

Povedzme, že sme si zvolili nasledovnú sadu identifikátorov:

číslo	identifikátor
0	6
1	2
2	9
3	3
4	7

Naša funkcia `label` potom musí na výstupe vrátiť pole `[6, 2, 9, 3, 7]`.



Počas druhého programu sa potom môže napríklad vyskytnúť nasledujúce volanie:

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

Toto znamená, že paket je v stanici s identifikátorom 9 a chce sa dostať do stanice s identifikátorom 6. Susedia stanice s identifikátorom 9 majú identifikátory 2 a 7.

Ak by sme sa dívali na pôvodné čísla vrcholov, potrebuje tento paket prejsť po ceste `[2, 1, 0]`. Ak sa dívame na ich identifikátory, ide o cestu `[9, 2, 6]`. Ďalší vrchol na tejto ceste má identifikátor 2. Naša funkcia má teda mať návratovú hodnotu 2.

Pozrime sa na ešte jedno možné volanie:

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

Toto volanie má vrátiť hodnotu 3, keďže vidíme, že adresátom tohto paketu je priamo jeden zo susedov aktuálneho vrcholu.

Obmedzenia

- $1 \leq r \leq 10$

Pre každé volanie `label` platí:

- $2 \leq n \leq 1000$
- $0 \leq u[i], v[i] \leq n - 1$ (pre $0 \leq i \leq n - 2$)

Pre každé volanie `find_next_station` platí:

- $1 \leq l \leq n - 1$
- $0 \leq s, t \leq k$
- $s \neq t$
- $0 \leq c[i] \leq k$ (pre $0 \leq i \leq l - 1$)
- $c[i - 1] < c[i]$ (pre $1 \leq i \leq l - 1$)
- Súčet dĺžok polí `c` vo všetkých volaniach funkcie `find_next_station` počas jedného behu programu neprekročí 100 000.

Podúlohy

1. (5 bodov) $k = 1000$ a každá stanica má nanajvýš dvoch susedov.
2. (8 bodov) $k = 1000$ a pre každé i platí, že linka i spája stanice $i + 1$ a $\lfloor \frac{i}{2} \rfloor$.
3. (16 bodov) $k = 1\,000\,000$ a existuje nanajvýš jedna stanica, ktorá má viac ako dvoch susedov
4. (10 bodov) $n \leq 8$, $k = 10^9$
5. (61 bodov) $k = 10^9$

V podúlohe 5 sa dá získať aj čiastočné skóre. Nech m je najväčšia hodnota identifikátora, ktorú ste naozaj použili (pričom maximum berieme cez úplne všetky volania vašej funkcie `label`). Vaše skóre za túto podúlohu sa potom bude počítať podľa nasledujúcej tabuľky:

Maximálny identifikátor	Skóre
$m \geq 10^9$	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

Ukážkový grader

Ukážkový grader očakáva vstup v nasledujúcom formáte:

- line 1: r
- blok i (pre $0 \leq i \leq r - 1$), popisujúci scenár číslo i :
 - line 1: n k
 - line $2 + i$ ($0 \leq i \leq n - 2$): $u[i]$ $v[i]$
 - line $1 + n$: číslo q udávajúce počet volaní funkcie `find_next_station`
 - line $2 + n + j$ ($0 \leq j \leq q - 1$): tri medzerou oddelené čísla predstavujúce jedno rútovanie: s , t a číslo toho suseda vrcholu číslo s , ktorý je najbližší ku t (a teda je správnou odpoveďou na túto otázku)

Ukážkový grader vypisuje nasledovné:

- line 1: m
- blok i (for all $0 \leq i \leq r - 1$), popisujúci scenár i :
 - line $1 + j$ ($0 \leq j \leq q - 1$): číslo stanice, ktorej identifikátor vrátila na výstupe `find_next_station`.

Dajte si pozor na to, že na rozdiel od skutočného testovania ukážkový grader pri jednom spustení robí volania oboch vašich funkcií.