

# Stations (stations)

Singapurška internetska okosnica (SIO) sastoji se od  $n$  stanica kojima su dodijeljeni **indeksi** od 0 do  $n - 1$ . Također postoji  $n - 1$  dvosmjernih veza, označenih cijelim brojevima od 0 do  $n - 2$ . Svaka veza spaja dvije različite stanice. Dvije stanice direktno povezane vezom nazivamo susjedima.

Put od stanice  $x$  do stanice  $y$  jest slijed različitih stanica  $a_0, a_1, \dots, a_p$ , takvih da je  $a_0 = x$ ,  $a_p = y$ , te je svaki uzastopni par stanica susjedan. Postoji **točno jedan** put od neke stanice  $x$  do neke druge stanice  $y$ .

Bilo koja stanica  $x$  može napraviti paket (podatkovni komad) i poslati ga bilo kojoj drugoj stanici  $y$  koju nazivamo **odredištem** paketa. Paket je potrebno navigirati kroz jedinstveni put od  $x$  do  $y$ . Razmotrimo stanicu  $z$  na kojoj se trenutno nalazi paket čije je odredište stanica  $y$  ( $z \neq y$ ). Stanica  $z$ :

1. izvršava **navigacijsku proceduru** koja određuje susjeda stanice  $z$  koji je na jedinstvenom putu od  $z$  do  $y$  i
2. proslijeđuje paket tom susjedu.

Nažalost, stanice nemaju veliku količinu memorije te ne mogu održavati popis svih veza singapurške internetske okosnice kako bi te podatke iskoristile u svojim navigacijskim procedurama.

Vaš je zadatak implementirati navigacijsku shemu SIO-a, koja se sastoji od dvije procedure.

- Prvoj je proceduri poznat  $n$ , popis svih veza SIO-a i cijeli broj  $k \geq n - 1$  preko ulaznih parametara. Ova procedura svakoj stanici pridodaje **jedinstvenu labelu** između 0 i  $k$ , uključivo
- Druga procedura jest navigacijska procedura koja je jednaka svim stanicama nakon određivanja labela prethodnom procedurom. Toj su proceduri poznate **isključivo** sljedeće vrijednosti:
  - $s$ , **labela** stanice kod koje se trenutno nalazi paket,
  - $t$ , **labela** odredišne stanice paketa ( $t \neq s$ ),
  - $c$ , lista **labela** svih susjeda stanice  $s$ .

navigacijska procedura treba vratiti **labelu** susjeda od  $s$  kojem je potrebno proslijediti paket.

Također, broj bodova vašeg rješenja u jednom podzadatku ovisi o maksimalnoj vrijednosti labela koja je dodijeljena nekoj stanici (po principu, manje je više).

## Implementacijski detalji

Potrebno je implementirati sljedeće procedure:

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

- $n$ : broj stanica singapurške internetske okosnice.
- $k$ : najveća moguća vrijednost labele.
- $u$  i  $v$ : polja duljine  $n - 1$  koja opisuju veze. Za svaki  $i$  ( $0 \leq i \leq n - 2$ ), veza  $i$  spaja stanice s indeksima  $u[i]$  i  $v[i]$ .
- Procedura treba vratiti polje  $L$  duljine  $n$ . Za svaki  $i$  ( $0 \leq i \leq n - 1$ )  $L[i]$  je labela dodijeljena stanici s indeksom  $i$ . Svi elementi polja  $L$  trebaju biti jedinstveni i između 0 i  $k$ , uključivo.

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

- $s$ : labela stanice kod koje se nalazi paket.
- $t$ : labela odredišne stanice paketa.
- $c$ : polje koje sadrži popis svih labela susjeda stanice  $s$ . Polje  $c$  sortirano je u rastućem poretku.
- Procedura treba vratiti labelu susjeda stanice  $s$  kojemu je potrebno proslijediti paket.

Svaki testni primjer sastoji se od jednog ili više nezavisnih scenarija (tj., različitih SIO-a). Za testni primjer koji sadrži  $r$  scenarija, **program** koji poziva gornje procedure poziva se točno dva puta, i to.

Tijekom prvog poziva programa:

- `label` procedura poziva se  $r$  puta,
- izlazne vrijednosti funkcije (labele) pohranjene su u ocjenjivačkom sustavu
- `find_next_station` se ne poziva.

Tijekom drugog poziva programa:

- `find_next_station` može biti pozvan više puta,
- labele predane svakom pozivu `find_next_station` su labele koje je vratio poziv procedure `label` za **proizvoljno odabran** scenarij tijekom prvog poziva.
- `label` se ne poziva.

Primijetite da se informacije pohranjene u statičkim ili globalnim varijablama prvog poziva programa ne vide prilikom pozivanja procedure `find_next_station` u drugom pozivu programa.

## Probni primjeri

Razmotrimo sljedeći poziv:

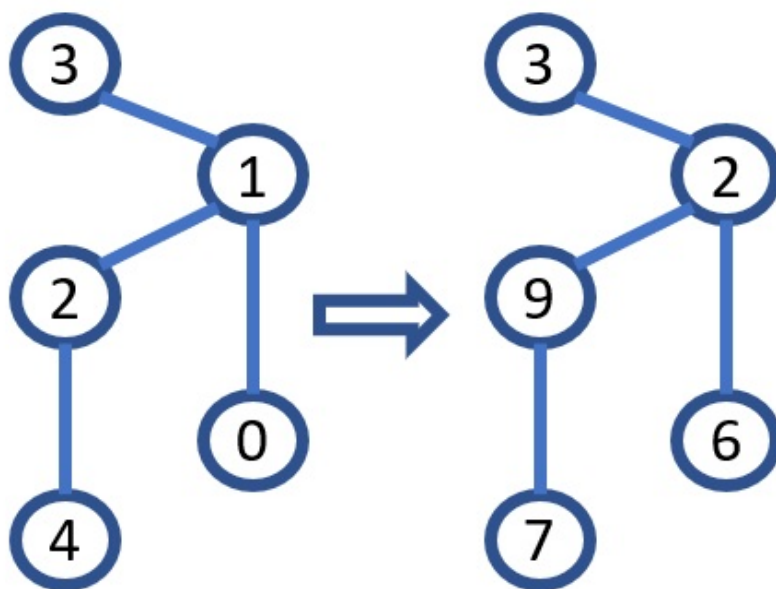
```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

Ukupno postoji 5 stanica i 4 veze koje povezuju stanice s indeksima (0, 1), (1, 2), (1, 3) i (2, 4). Svaka labela može biti cijeli broj između 0 i  $k = 10$ .

Da bismo stanicama dodijelili sljedeće labele:

Indeks	Labela
0	6
1	2
2	9
3	3
4	7

procedura `label` treba vratiti `[6, 2, 9, 3, 7]`. Na sljedećoj slici, lijeva strana predstavlja indekse stanica, a desna predstavlja labele.



Pretpostavimo da su labele dodijeljene kako je gore opisano te razmotrimo sljedeći poziv:

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

Vidimo da stanica koja sadrži paket ima oznaku 9 te da je odredšite paketa u stanici s labelom 6. Labele stanica na putu do odredišne stanice su `[9, 2, 6]`. Stoga, procedura treba vratiti 2.

Razmotrimo još jedan poziv:

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

Procedura treba vratiti 3, jer odredišna stanica paketa ima labelu 3, a upravo je ona susjed stanice s labelom 2. Stoga, potrebno je direktno njoj preusmjeriti paket.

## Constraints

- $1 \leq r \leq 10$

Za svaki poziv procedure `label`:

- $2 \leq n \leq 1000$
- $k \geq n - 1$
- $0 \leq u[i], v[i] \leq n - 1$  (za svaki  $0 \leq i \leq n - 2$ )

Za svaki poziv procedure `find_next_station`, uz pretpostavku da je  $d$  duljina polja  $c$ :

- $1 \leq d \leq n - 1$
- $0 \leq s, t \leq k$
- $s \neq t$
- $0 \leq c[i] \leq k$  (za svaki  $0 \leq i \leq d - 1$ )
- $c[i - 1] < c[i]$  (za svaki  $1 \leq i \leq d - 1$ )
- Ukupna duljina svih polja  $c$  prosljeđenih proceduri `find_next_station` ne premašuje 100 000 zbirno po svim scenarijima zajedno.

## Podzadaci

1. (5 bodova)  $k = 1000$ , niti jedna stanica nema više od 2 susjeda.
2. (8 bodova)  $k = 1000$ , veza  $i$  povezuje stanice  $i + 1$  i  $\lfloor \frac{i}{2} \rfloor$ .
3. (16 bodova)  $k = 1\,000\,000$ , najviše jedna stanica ima više od 2 susjeda.
4. (10 bodova)  $n \leq 8$ ,  $k = 10^9$
5. (61 bod)  $k = 10^9$

U podzadatku 5 možete osvojiti parcijalne bodove. Neka je  $m$  najveća labela koju vraća procedura `label` po svim scenarijima. Vaši bodovi za ovaj podzadatak računaju se pomoću sljedeće tablice:

Maksimalna labela	Bodovi
$m \geq 10^9$	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

## Ogledni ocjenjivač

Ogledni ocjenjivač čita ulaz u sljedećem formatu:

- redak 1:  $r$

Slijedi  $r$  blokova, svaki opisuje jedan scenarij. Format svakog bloka jest:

- redak 1:  $n \ k$
- redak  $2 + i$  ( $0 \leq i \leq n - 2$ ):  $u[i] \ v[i]$
- redak  $1 + n$ : broj poziva procedure `find_next_station`,  $q$ .
- redak  $2 + n + j$  ( $0 \leq j \leq q - 1$ ):  $z[j] \ y[j] \ w[j]$ : **indeksi** stanica uključenih u  $j$ -ti poziv procedure `find_next_station`: stanica  $z[j]$  sadrži paket, stanica  $y[j]$  je odredišna stanica, a stanica  $w[j]$  je stanica koja slijedi nakon  $z[j]$  na jedinstvenom putu od  $z[j]$  do  $y[j]$ .

Ogledni ocjenjivač ispisuje rezultat u sljedećem formatu:

- redak 1:  $m$

Slijedi  $r$  blokova koji odgovaraju scenarijima iz ulaza. Format svakog bloka jest:

- redak  $1 + j$  ( $0 \leq j \leq q - 1$ ): **indeks** stanice čija **labela** je bila izlazna vrijednost  $j$ -tog poziva procedure `find_next_station` u tom scenariju.

Primijetite da svaki poziv oglednog ocjenjivača poziva i `label` i `find_next_station`.