Stations (stations)

Singapore's Internet Backbone (SIB) sastoji se od n stanica, kojima su dodijeljeni **indeksi** od 0 do $n-1$. Postoji i $n-1$ dvosmjernih linkova, označenih brojevima od 0 do $n-2$. Svaki link povezuje dvije različite stanice. Dvije stanice koje povezuje jedan link su susjedne.

Putanja od stanice x do stanice y je niz različitih stanica $a_0, a_1, \cdots, a_p$, tako da $a_0 = x$, $a_p = y$, i svake dvije uzastopne stanice u putanji su susjedi. Postoji **tačno jedna** putanja od bilo koje stanice x do neke druge stanice y.

Svaka stanica x može kreirati paket podataka i poslati ga drugoj stanici y; tada se stanica y naziva **odredištem (ciljem) paketa**. Paket se šalje kroz jedinstvenu putanju od x do y na sljedeći način. Posmatrajmo stanicu z u kojoj je trenutno paket čije je odredište stanica y ($z \neq y$). U ovoj situaciji stanica z:

1. izvršava **proceduru rutiranja** tj. određuje susjeda stanice z koji je na jedinstvenoj stazi od z do y, i
2. prosljeđuje paket susjedu.

Međutim, stanice imaju ograničenu memoriju i ne čuvaju kompletnu listu linkova u SIB-u kako bi se mogle iskoristiti u proceduri rutiranja.

Važ zadatak je da implementirate šemu rutiranja za SIB koja se sastoji od dvije funkcije

- Prva funkcija daje n, listu linkova u SIB i cio broj $k \geq n-1$ kao ulaze. Svakoj stanici dodjeljuje **jedinstvenu** cjelobrojnu **labelu** između 0 i k, uključujući i k.
- Druga funkcija je ruting funkcija, koja se raspoređuje na sve stanice nakon dodjeljivanja labela. Dati su **samo** sljedeći ulazi:
    - s, **labela** stanice koja trenutno drži paket,
    - t, **labela** ciljne stanice paketa ($t \neq s$),
    - c, lista **labela** svih susjeda stanice s.

    Trebalo bi da vrati **labelu** susjeda s na koji paket treba da se proslijedi.

U jednom podzadatku, rezultat vašeg rješenja zavisi od vrijednosti maksimalne labele dodijeljene bilo kojoj stanici (generalno, manje je bolje).

Detalji implementacije

Potrebno je implementirati sljedeću funkciju:

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

- n: broj stanica u SIB-u.
- k: maksimalna labela koje će se koristiti.
- u i v: nizovi veličine $n-1$ koji opisuju linkove. Za svaki i ($0 \leq i \leq n-2$), link i povezuje stanice sa indeksima $u[i]$ i $v[i]$.
- Ova funkcija treba da vrati jedan niz L veličine n. Za svaki i ($0 \leq i \leq n-1$) $L[i]$ je labela dodijeljena stanici sa indeksom i. Svi elementi niza L moraju biti jedinstveni i moraju biti između 0 i k, uključujući k.

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

- s: labela stanice koja drži paket.
- t: labela ciljne stanice paketa.
- c: niz koji daje listu labela svih susjeda stanice s. Niz c sortiran je u rastućem redosljedu.
- Ova funkcija treba da vrati labelu susjeda stanice s na koji treba proslijediti paket.

Svaki test uključuje jedan ili više nezavisnih scenarija (tj. različite SIB opise). Za test slučaj koji uključuje r scenarija, **program** koji poziva gornje funkcije pokreće se tačno dva puta, kako slijedi.

Tokom prvog pokretanja programa:

- funkcija `label` poziva se r puta,
- vraćene labele čuva program za ocjenjivanje (grader), i
- `find_next_station` se ne poziva.

Tokom drugog izvođenja programa:

- `find_next_station` može se pozivati više puta. U svakom pozivu bira se **proizvoljan** scenario, pa se onda labele koje je vratio poziv funkcije `label` za taj scenario koriste kao ulaz za `find_next_station`.
- `label` nije pozivan

Konkretno, sve informacije sačuvane u statičkim ili globalnim promenljivim u prvom pokretanju programa nisu dostupne unutar funkcije `find_next_station`.

Primjer

Razmotrite sljedeći poziv:

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

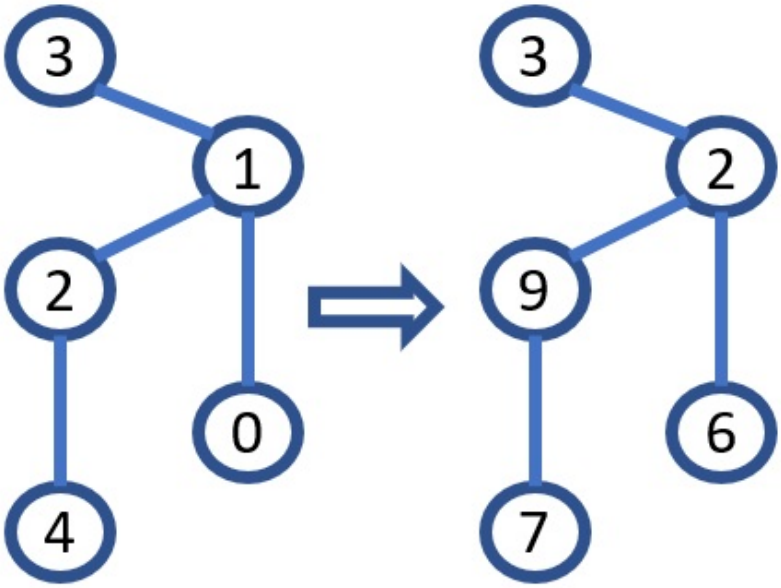
Ukupno ima 5 stanica i 4 linka koji povezuju parove stanica sa indeksima $(0, 1)$, $(1, 2)$, $(1, 3)$ i

$(2, 4)$. Svaka labela može biti cijeli broj od 0 do $k = 10$.

Da biste prijavili sledeće označavanje:

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

funkcija `label` vraća $[6, 2, 9, 3, 7]$. Brojevi na sljedećoj slici prikazuju indekse (lijevi panel) i dodijeljene labele (desni panel).



Pretpostavimo da su labele dodijeljene kako je gore opisano i razmotrite sljedeći poziv:

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

To znači da stanica koja drži paket ima labelu 9, a odredišna stanica ima labelu 6. Labele stanica na putu do ciljne stanice su $[9, 2, 6]$. Stoga poziv treba da vrati 2, što je labela stanice na koju paket treba da se proslijedi (koja ima indeks 1).

Razmotrite još jedan mogući poziv:

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

Funkcija treba da vrati 3, jer je ciljna stanica sa labelom 3 susjed stanice sa labelom 2, i stoga treba

da primi paket direktno.

Ograničenja

- $1 \leq r \leq 10$

Za svaki poziv `label`:

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

Za svaki poziv `find_next_station`, ulaz dolazi iz proizvoljno izabranog prethodnog poziva `label`. Posmatrajte labele koje je proizveo ovaj poziv. Tada:

- s i t su labele dvije različite stanice.
- c je niz labela svih susjeda stanice sa labelom s, u rastućem redosljedu.

Za svaki test slučaj, ukupna dužina svih nizova c proslijeđenih funkciji `find_next_station` ne prelazi $100\ 000$ za sve scenarije zajedno.

Podzadaci

1. (5 bodova) $k = 1000$, nijedna stanica nema više od 2 susjeda.
2. (8 bodova) $k = 1000$, link i povezuje stanice $i + 1$ i $\left\lfloor \frac{i}{2} \right\rfloor$.
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 broj bodova zavisi od vrijednosti m, gdje je m maksimalna vrijednost labele koju vraća funkcija `label` u svim scenarijima. Rezultat za ovaj podzadatak izračunava se prema sljedećoj tabeli:

Max. vrijednost labele	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

Program za ocjenjivanje (grader)

Program za ocjenjivanje (grader) čita ulaz u sljedećem formatu:

- red 1: r

Zatim slijedi r blokova, gdje svaki blok opisuje jedan scenario. Format svakog bloka je sljedeći:

- red 1: $n \ \ k$
- red $2 + i$ ($0 \leq i \leq n - 2$): $u[i] \ \ v[i]$
- red $1 + n$: q: broj poziva `find_next_station`.
- red $2 + n + j$ ($0 \leq j \leq q - 1$): $z[j] \ \ y[j] \ \ w[j]$: **indeksi** stanica uključenih u j-ti poziv `find_next_station`. Stanica $z[j]$ drži paket, stanica $y[j]$ je cilj paketa, a stanica $w[j]$ je stanica kojoj paket treba da bude proslijeđen.

Program za ocjenjivanje (grader) štampa rezultat u sljedećem formatu:

- red 1: m

Slijedi r blokova koji odgovaraju uzastopnim scenarijima na ulazu. Format svakog bloka je sljedeći:

- red $1 + j$ ($0 \leq j \leq q - 1$): **indeks** stanice, čija je **labela** vraćena j-tim pozivom `find_next_station` u ovom scenariju.

Imajte na umu da svako pokretanje programa za ocjenjivanje (gradera) poziva i `label` i `find_next_station`.