Stations (stations)

Singapurning "Singapore's Internet Backbone (SIB)" tizimi 0 dan $n-1$ gacha **indekslangan** n ta stansiyadan iborat. Shuningdek $n-1$ ta ikki tomonlamali simlar ham mavjud bo'lib, ular 0 dan $n-2$ gacha raqamlangan. Har bir sim ikkita turli xil stansiyani bog'laydi. Bitta sim bilan bog'langan ikkita stansiyalar qo'shnilar deyiladi.

x stansiyadan y stansiyagcha bo'lgan yo'l deb shunaqangi $a_0, a_1, \cdots, a_p$ stansiyalar ketma-ketligiga aytiladiki, $a_0 = x$, $a_p = y$ va yo'ldagi barcha ketma-ket kelgan stansiyalar qo'shni stansiyalar. Ixtiyoriy x stansiyadan y stansiyaga aniq bitta yo'l bor.

Ixtiyoriy x stansiya, ma'lumot yaratishi va uni ixtiyoriy boshqa y stansiyaga jo'natishi mumkin va buni ma'lumotni **borishi kerak bo'lgan nuqtasi** deyiladi. Bu ma'lumot x dan y ga quyidagicha borishi lozim. Hozirda m'alumotni o'zida saqlab turgan z nuqtani ko'raylik, uni borishi kerak bo'lgan nuqtasi y ($z \neq y$). Bu holda z:

1. **marshrutlash amalini** bajaradi va u z dan y gacha bo'lgan yagona yo'lda joylashgan z ni qo'shnisini topib beradi.
2. ma'lumotni ni shu qo'shniga uzatadi.

Ammo, stansiyalarda xotira cheklangan va ular SIB dagi barcha bog'lanishlarni saqlamaydi.

Sizning vazifangiz ikkita funksiyadan iborat bo'lgan SIB ni marshrutlash sxemasini bajarishdan iborat:

- Birinchi funksiyada n, SIB dagi simlar ro'yxati va $k \geq n-1$ soni beriladi. U har bir stansiyaga 0 dan k gacha oralig'ida bo'lgan **turli xil** sonlardan foydalanib **raqamlaydi**.
- Ikkinchi funksiya marshrutlash funksiyasi bo'lib, u raqamlash funksiyasidan keyin ishga tushiriladi. Unda **faqatgina** quyidagi kiruvchi ma'lumotlar bor:
  - s, hozirda ma'lumotni ushlab turgan stansiya **raqami**,
  - t, ma'lumotni borishi kerak bo'lgan nuqtasi **raqami** ($t \neq s$),
  - c, s ning barcha qo'shinlarini **raqamlari** ro'yxati.

Bu funksiya ma'lumot jo'natilishi kerak bo'lgan **s** stansiyani qo'shnisini **raqamini** qaytarishi lozim.

Qism masalalarning birida, siz oladigan ball stansiyalarga qo'yadigan raqamlaringizga bog'liq(ya'ni, qancha kichik bo'lsa shuncha yaxshi).

Tafsilotlar

Quyidagi funksiyalarni bajarishingiz lozim:

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

- n: SIBdagi stansiyalar soni.
- k: raqamlashda ishlatilishi mumkin bo'lgan raqamlar soni.
- u va v: Simlarni ifodalaydigan uzunligi $n-1$ bo'lgan massiv. Har bir i ($0 \le i \le n-2$) uchun, i-sim $u[i]$ va $v[i]$ stansiyalarni bog'laydi.
- Bu funksiya uzunligi n bo'lgan bitta L massivni qaytarishi lozim. Har bir i ($0 \le i \le n-1$) uchun $L[i]$ i-indexli stansiyaga belgilangan raqamni bildiradi. Massivni har bir elementlari 0 va k oralig'idagi turli xil sonlar bo'lishi lozim.

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

- s: ma'lumotni ushlab turgan stansiya raqami.
- t: Ma'lumot borishi kerak bo'lgan stansiya raqami.
- c: s ni barcha qo'shnilarini ifodalovchi raqamlar ro'yxati. c massiv o'suvchi tartibda saralangan
- Bu funksiya ma'lumot jo'natilishi kerak bo'lgan s ning qo'shnisini qaytarishi lozim.

Har bir test bir yoki bir nechta testlar to'plamidan iborat bo'lishi mumkin(Turli xil SIB ko'rinishlari). r ta testlar to'plamidan iborat bo'lgan testda, yuqoridagi funksiyalarni chaqiruvchi **dastur** quyidagicha ikki marta chairiladi.

Birinchi chaqiruvda:

- `label` funksiyasi `r` marta chaqiriladi,
- qaytarilgan raqamlashlar baholash tizimi yordamida yuklab olinadi va
- `find_next_station` chaqirilmaydi.

Ikkinchi chaqiruvda:

- `find_next_station` ko'p marta chaqirilishi mumkin. Har bir chaqiruvda **ixtiyoriy** test to'plami tanlanadi va `label` funksiyasida berilgan raqamlashdan foydalaniladi.
- `label` chaqirilmaydi.

Boshqacharoq aytkanda, static yoki global ravishda saqlangan o'zgaruvchilar, `find_next_station`ni keyingi chaqiruvlarida mavjud bo'lmaydi.

Namuna

Quyidagi chaqiruvni olaylik:

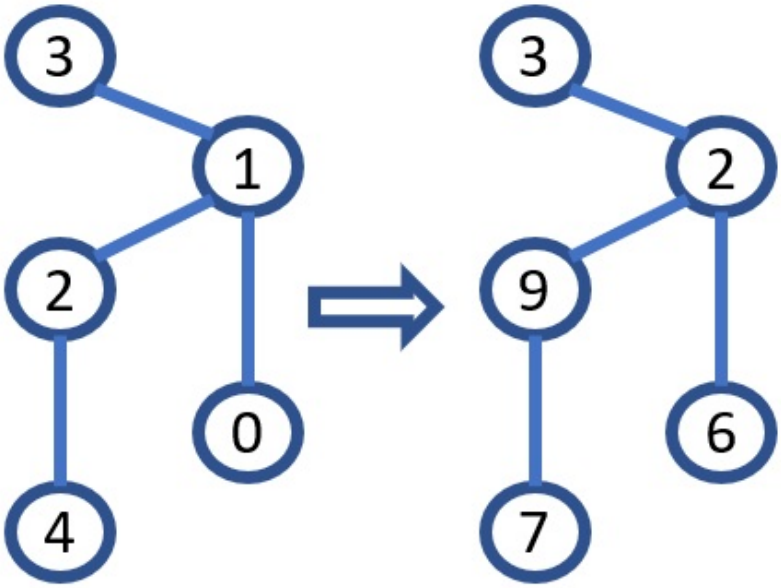
```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

There are a total of 5 stations, and 4 links connecting pairs of stations with indices $(0,1)$, $(1,2)$, $(1,3)$ and $(2,4)$. Each label can be an integer from 0 to $k=10$.

In order to report the following labelling:

Index	Label
0	6
1	2
2	9
3	3
4	7

the `label` procedure should return [6, 2, 9, 3, 7]. The numbers in the following figure show the indices (left panel) and assigned labels (right panel).



Assume the labels have been assigned as described above and consider the following call:

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

This means that the station holding the packet has label 9, and the target station has label 6. The labels of stations on the path to the target station are $[9, 2, 6]$. Hence, the call should return 2, which is the label of the station that the packet should be forwarded to (which has index 1).

Consider another possible call:

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

The procedure should return 3, since the target station with label 3 is a neighbour of the station with label 2, and hence should receive the packet directly.

Cheklovlar

- $1 \leq r \leq 10$

For each call to `label`:

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

For each call to `find_next_station`, the input comes from an arbitrarily chosen previous call to `label`. Consider the labels it produced. Then:

- s and t are labels of two different stations.
- c is the sequence of all labels of neighbours of the station with label s, in ascending order.

For each test case, the total length of all arrays c passed to the procedure `find_next_station` does not exceed $100\ 000$ for all scenarios combined.

Qism masalalar

1. (5 points) $k = 1000$, no station has more than 2 neighbours.
2. (8 points) $k = 1000$, link i connects stations $i + 1$ and $\left\lfloor \frac{i}{2} \right\rfloor$.
3. (16 points) $k = 1\ 000\ 000$, at most one station has more than 2 neighbours.
4. (10 points) $n \leq 8$, $k = 10^9$
5. (61 points) $k = 10^9$

In subtask 5 you can obtain a partial score. Let m be the maximum label value returned by `label` across all scenarios. Your score for this subtask is calculated according to the following table:

Maximum label	Score
$m \geq 10^9$	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

Grader

The sample grader reads the input in the following format:

- line 1: r

r blocks follow, each describing a single scenario. The format of each block is as follows:

- line 1: $n \;\; k$
- line $2+i$ ($0 \le i \le n-2$): $u[i] \;\; v[i]$
- line $1+n$: q: the number of calls to `find_next_station`.
- line $2+n+j$ ($0 \le j \le q-1$): $z[j] \;\; y[j] \;\; w[j]$: **indices** of stations involved in the j-th call to `find_next_station`. The station $z[j]$ holds the packet, the station $y[j]$ is the packet's target, and the station $w[j]$ is the station that the packet should be forwarded to.

The sample grader prints the result in the following format:

- line 1: m

r blocks corresponding to the consecutive scenarios in the input follow. The format of each block is as follows:

- line $1+j$ ($0 \le j \le q-1$): **index** of the station, whose **label** was returned by the j-th call to `find_next_station` in this scenario.

Note that each run of the sample grader calls both `label` and `find_next_station`.