Станции (stations)

Интернет гръбнака на Сингапур (ИГС) се състои от n станции, на които са зададени **индекси** от 0 до $n-1$. Също така има $n-1$ двупосочни връзки, номерирани от 0 до $n-2$. Всяка връзка свързва две различни станции. Две станции, свързани с пряка връзка, се наричат съседни.

Път от станция x до станция y е последователност от различни станции $a_0, a_1, \cdots, a_p$, така че $a_0 = x$, $a_p = y$ и всеки две съседни станции в пътя са съседни. Има **точно един** път от всяка станция x до всяка друга станция y.

Всяка станция x може да създаде пакет (пакет от данни) и да го изпрати на някоя друга станция y, която се нарича **цел** на пакета. Този пакет трябва да бъде насочен по единствения път от x до y както следва. Нека станция z в момента държи пакет, чиято цел е станция y ($z \neq y$). В тази ситуация станция z:

1. изпълнява **насочваща процедура**, която определя съседа на z, който е на единствения път от z до y и
2. насочва пакета към този съсед.

Станциите обаче имат ограничена памет и не съхраняват целия списък с връзките в ИГС, за да го използват в насочващата процедура.

Вашата задача е да направите насочваща схема за ИГС, която се състои от две функции.

- Първата функция е по дадено n, списък с връзките в ИГС и цяло число $k \geq n-1$, като входни данни. Тя задава на всяка станция **различен етикет**, който е цяло число между 0 и k, включително.

- Втората функция е насочваща процедура, която се разполага във всички станции след присвояването на етикети. Тя получава **само** следните входи данни:

  - s, **етикетът** на станцията, която държи пакета в момента,
  - t, **етикетът** на станцията цел на пакета ($t \neq s$),
  - c, списъкът с **етикетите** на всички съседи на s.

  Тя трябва да върне **етикетът** на съседа на s към който пакетът трябва да бъде насочен.

В една подзадача, резултатът на вашето решение зависи от най-голямата стойност на етикет даден на някоя станция (общо взето, по-малкото е по-добре).

Имплементация

Трябва да напишете следните функции:

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

- n: броят станции в ИГС.
- k: най-големият етикет, който може да бъде използван.
- u и v: масиви с размер $n-1$, описвайки връзките. За всяко i ($0 \leq i \leq n-2$) връзка i свързва станции с индекси $u[i]$ и $v[i]$.
- Тази функция трябва да върне един масив L с размер n. За всяко i ($0 \leq i \leq n-1$) $L[i]$ е етикатът даден на станция с индекс i. Всички елементи на масива L трябва да са различни числа между 0 и k, включително.

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

- s: етикет на станция държаща пакет.
- t: етикет на станция, която е целта на пакета.
- c: масив съдържащ списък с етикетите на всички съседи на s. Масивът c е подреден в нарастващ ред.
- Тази функция трябва да върне етикета на съседа на s, към който пакетът трябва да бъде насочен.

Всеки тест включва един или повече независими сценарии (т.е., описания на различни ИГС). За тест включващ r сценария, **програма** извикваща горните функции се пуска точно два пъти, както следва.

По време на първото изпълнение на програмата:

- функцията `label` се извиква r пъти,
- върнатите етикети се запазват от оценяващата система и
- функцията `find_next_station` не се извиква.

По време на второто изпълнение на програмата:

- `find_next_station` може да бъде извикана много пъти. При всяко извикване, се избира **произволен** сценарий и етикетите върнати от извикването на функцията `label` в този сценарий се използват като входни данни на `find_next_station`.
- `label` не се извиква.

В частност, информация запазена в статични или глобални променливи по време на първото изпълнение на програмата не е на разположение по време на изпълнение на функията `find_next_station`.

Примери

Разглеждаме следното извикване:

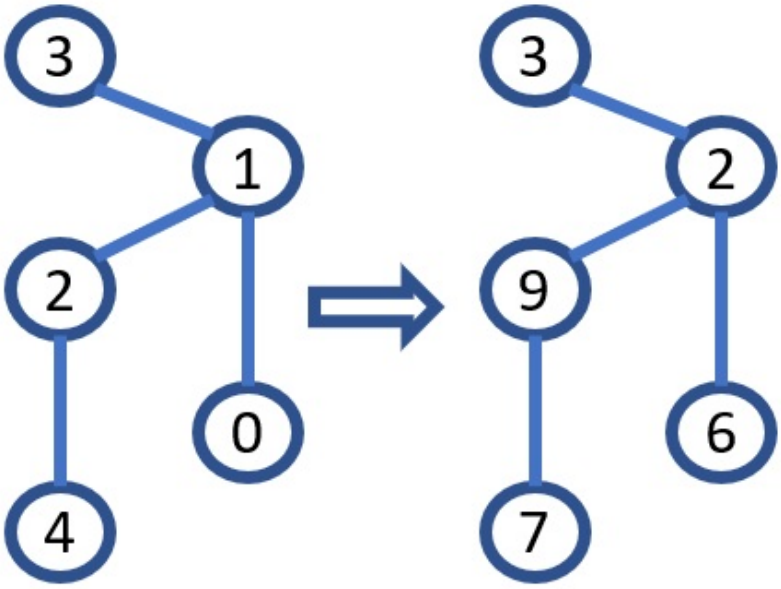
```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

Има общо 5 станции и 4 връзки свързващи двойки станции с индекси $(0, 1)$, $(1, 2)$, $(1, 3)$ и $(2, 4)$. Всеки етикет може да бъде цяло число от 0 до $k = 10$.

За да съобщим за следното етикетиране:

Индекс	Етикет
0	6
1	2
2	9
3	3
4	7

функцията `label` трябва да върне $[6, 2, 9, 3, 7]$. Числата на фигурата погазват индексите (в лявата част) и етикетите (в дясната част).



Приемете, че етикетите са дадени както е описано по-горе и разглеждаме следното извикване:

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

Това означава, че станцията която държи пакета има етикет 9 и станцията цел има етикет 6. Етикетите на станциите, на пътя към станцията цел, са $[9, 2, 6]$. Следователно, функцията трябва да върне 2, което е етикетът на станцията към която пакетът трябва да бъде насочен (която има индекс 1).

Разглеждаме друго възможно извикване:

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

Тази функция трябва да върне 3, понеже станцията цел с етикет 3 е съсед на станцията с етикет 2 и значи трябва да получи пакета директно.

Ограничения

- $1 \leq r \leq 10$

За всяко извикване на `label`:

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

За всяко извикване на `find_next_station`, входните данни идват от произволно избрано предишно извикване на `label`. Разглеждаме етикетите, които са върнати. Тогава:

- s и t са етикетите на две различни станции.
- c е списък с всички етикети на съседи на станцията с етикет s, в нарастващ ред.

За всеки тест, общата дължина на всички масиви c, подадени на функцията `find_next_station`, не надвишава $100\ 000$ сумарно за всички сценарии.

Подзадачи

1. (5 точки) $k = 1000$, няма станция с повече от 2 съседа.
2. (8 точки) $k = 1000$, връзка i свързва станции $i + 1$ и $\left\lfloor \frac{i}{2} \right\rfloor$.
3. (16 точки) $k = 1\ 000\ 000$, най-много една станция има повече от 2 съседа.
4. (10 точки) $n \leq 8$, $k = 10^9$
5. (61 точки) $k = 10^9$

В подзадача 5 може да получите частичен резултат. Нека m е най-голямата стойност на етикет, върната от `label`, измежду всички сценарии. Вашият резултат за тази подзадача се изчислява по следната таблица:

Най-голям етикет	Резултат
$m \geq 10^9$	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

Примерен грейдър

Примерният грейдър чете входа в следния формат:

- ред 1: r

следват r блока, всеки описващ един сценарий. Форматът на всеки блок е както следва:

- ред 1: $n \ \ k$
- ред $2 + i$ $(0 \le i \le n - 2)$: $u[i] \ \ v[i]$
- ред $1 + n$: q − броят извиквания на `find_next_station`.
- ред $2 + n + j$ $(0 \le j \le q - 1)$: $z[j] \ \ y[j] \ \ w[j]$ − **индекси** на станциите, участващи в j-тото извикване на `find_next_station`. Станцията $z[j]$ държи пакета, станцията $y[j]$ е целта на пакета и $w[j]$ е станцията, към която пакетът трябва да бъде насочен.

Примерният грейдър отпечатва изхода в следния формат:

- ред 1: m

следват r блока, съответстващи на последователните сценарии във входа. Форматът на всеки блок е както следва:

- ред $1 + j$ $(0 \le j \le q - 1)$: **индекс** на станцията, чийто **етикет** е върнат от j-тото извикване на `find_next_station` в този сценарий.

Забележете, че всяко извикване на примерния грейдър извиква и `label`, и `find_next_station`.