

ایستگاهها (stations)

زیرساخت اینترنت سنگاپور (SIB) شامل n ایستگاه است که با اندیس‌های 0 تا $1 - n$ شماره گذاری شده‌اند. علاوه بر این، $1 - n$ لینک بدون جهت با شماره‌های 0 تا $2 - n$ وجود دارد. هر لینک دو ایستگاه متمایز را به هم وصل می‌کند. دو ایستگاه که با لینک به یکدیگر متصل هستند را همسایه می‌نامیم.

یک مسیر از ایستگاه x به ایستگاه y دنباله‌ای از ایستگاه‌های a_p, a_1, \dots, a_0 است که $a_0 = x$ ، $a_p = y$ و هر دو ایستگاه مجاوری در این مسیر، همسایه هستند. بین هر دو ایستگاهی مانند x و y دقیقاً یک مسیر وجود دارد.

هر ایستگاه مانند x می‌تواند یک بسته (قطعه‌ای از داده) ایجاد کند و به هر ایستگاه دیگری مانند y که مقصد بسته نامیده می‌شود، بفرستد. این بسته باید از مسیر یکتاوی که از x به y وجود دارد حرکت کند. ایستگاه z را در نظر بگیرید که در حال حاضر بسته‌ای با مقصد y دارد. در این وضعیت، ایستگاه z باید:

1. یک **رویه مسیریابی** را اجرا کند که این رویه، همسایه‌ای از z که در مسیر یکتا از z به y هست را مشخص کند.
2. بسته را به این همسایه بفرستد.

البته ایستگاه‌ها حافظه محدودی دارند و نمی‌توانند تمام لینک‌های SIB را برای رویه مسیریابی نگهداری کنند.

وظیفه شما این است که یک برنامه مسیریابی برای SIB طراحی کنید که شامل دو تابع زیر باشد.

- تابع اول، n ، لیست لینک‌های SIB و عدد طبیعی $1 - k \geq n$ را به عنوان ورودی می‌گیرد. این تابع به هر ایستگاه یک لیل صحیح و یکتا بین 0 تا k (شامل این دو عدد) نسبت می‌دهد.
- تابع دوم رویه مسیریابی را پیاده‌سازی می‌کند. این تابع بر روی تمام ایستگاه‌ها، بعد از این‌که لیل‌هایشان مشخص شود، مستقر می‌شود. به این تابع تنها ورودی‌های زیر داده می‌شود:
 - s ، لیل ایستگاه فعلی که بسته را نگه داشته است.
 - t ، لیل ایستگاه مقصد بسته ($s \neq t$).
 - c ، لیستی از **لیل‌های همسایه‌های s** .

این تابع باید **لیل همسایه‌ای از s** که بسته باید به آن فرستاده شود را برگرداند.

در یک زیرمسئله، نمره شما به مقدار بزرگترین لیلی که به هر ایستگاهی نسبت داده‌اید، بستگی دارد (در مجموع، هر چه این مقدار کمتر باشد بهتر است).

جزئیات پیاده‌سازی

شما باید تابع زیر را پیاده‌سازی کنید:

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

- n : تعداد ایستگاهها در SIB.
- k : بزرگترین لبیلی که می‌توان استفاده کرد.
- u و v : آرایه‌هایی به طول $1 - n$ که لینک‌ها را مشخص می‌کنند. برای هر i ($0 \leq i \leq n - 2$), لینک i ایستگاه‌هایی با اندیس $[i]u$ و $[i]v$ را به هم متصل می‌کند.
- این تابع باید آرایه L به طول n را برگرداند. برای هر i ($0 \leq i \leq n - 1$), $[i]L$ لبیلی است که به ایستگاه i نسبت داده شده است. تمام اعضای آرایه L باید بین 0 تا k (شامل این دو عدد) و یکتا باشند.

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

- s : لبیل ایستگاهی که بسته را نگه داشته است.
- t : لبیل ایستگاه مقصد بسته.
- c : آرایه‌ای که لیست لبیل‌های تمام همسایه‌های s است. آرایه c به صورت صعودی مرتب شده است.
- این تابع باید لبیل همسایه‌ای از s که بسته باید به آن فرستاده شود را برگرداند.

هر تست کیس شامل یک یا تعدادی سناریو مستقل می‌شود (در واقع شامل تعدادی ساختار متفاوت برای SIB است). برای هر تست کیس شامل ۲ سناریو، یک برنامه که تابع بالا را صدا می‌زند دقیقاً دوبار به صورت زیر اجرا می‌شود.

طی اجرای اول برنامه:

- تابع `label`, r مرتبه صدا زده می‌شود.
- لبیل‌های برگردانده شده توسط سیستم ارزیاب ذخیره می‌شوند.
- تابع `find_next_station` صدا زده نمی‌شود.

طی اجرای دوم برنامه:

- تابع `find_next_station` ممکن است چندین مرتبه صدا زده شود. در هر مرتبه، یک سناریو دلخواه انتخاب می‌شود و لبیل‌هایی که در تابع `label` برگردانده شده‌اند، به عنوان ورودی در تابع `find_next_station` استفاده می‌شوند.
- تابع `label` صدا زده نمی‌شود.

در واقع اطلاعاتی که در متغیرهای `global` و `static` در اجرای اول ذخیره شده‌اند، در تابع `find_next_station` در دسترس نمی‌باشند.

مثال

فراخوانی زیر را در نظر بگیرید:

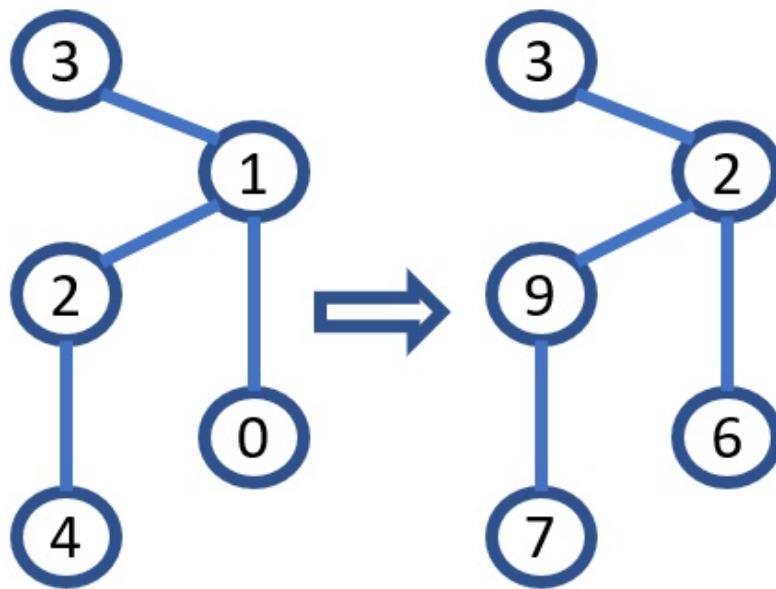
```
label(5, 10, [0, 1, 1, 2], [1, 2, 3, 4])
```

در مجموع ۵ ایستگاه وجود دارد که ۴ لینک اندیس‌های $(0, 1)$, $(1, 2)$ و $(2, 4)$ را به هم متصل کرده است. هر لبیل می‌تواند عدد صحیحی بین ۰ تا $10 = k$ باشد.

برای اینکه لبیل گذاری زیر را برگردانیم:

اندیس	لیبل
0	6
1	2
2	9
3	3
4	7

تابع `label` باید $[7, 3, 2, 9, 6]$ را برگرداند. در تصویر زیر اندیس‌ها (در تصویر چپ) و لیل‌های نسبت داده شده (در تصویر راست) قابل مشاهده است.



فرض کنید که لیل‌ها به صورتی که در بالا گفته شده، نسبت داده شده‌اند و فراخوانی زیر صورت می‌گیرد:

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

بدین معنا است که ایستگاهی که بسته را نگه داشته است لیل 9 و ایستگاه مقصد نیز لیل 6 دارد. لیل ایستگاه‌هایی که در مسیر به ایستگاه مقصد هستند، $[9, 2, 6]$ است. از این رو، فراخوانی باید 2 برگرداند که لیل ایستگاهی است که بسته باید به آن فرستاده شود (که اندیس 1 دارد).

فراخوانی زیر را در نظر بگیرید:

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

این تابع باید 3 برگرداند. زیرا مقصد لیل 3 دارد که همسایه ایستگاه با لیل 2 است و باید بسته را به صورت مستقیم دریافت کند.

محدودیت‌ها

$$1 \leq r \leq 10$$

برای هر فراخوانی `label`:

$$\begin{aligned} 2 \leq n \leq 1000 & \bullet \\ k \geq n - 1 & \bullet \\ (0 \leq i \leq n - 2 \text{ و } 0 \leq u[i], v[i] \leq n - 1) & \bullet \end{aligned}$$

برای هر فراخوانی تابع `find_next_station`, ورودی از یک فراخوانی دلخواهی که قبل روی `label` داشته‌ایم، می‌آید. لیل‌هایی که در این سناریو تولید شده است را در نظر بگیرید. آنگاه:

- s و t لیل‌های دو ایستگاه متفاوت هستند.
- c دنباله‌ای از لیل‌های همسایه‌های ایستگاه s به ترتیب صعودی است.

برای تمام تست‌کیس‌های مجموع کل سناریوها، مجموع طول آرایه c که به تابع `find_next_station` داده می‌شود از 100 000 عبور نمی‌کند.

زیرمسئله‌ها

- .1 (5 امتیاز) $k = 1000$, هیچ ایستگاهی بیش از 2 همسایه ندارد.
- .2 (8 امتیاز) $k = 1000$, لینک i ایستگاه $i + 1$ و $\lfloor \frac{i}{2} \rfloor$ را به هم متصل می‌کند.
- .3 (16 امتیاز) $k = 1\,000\,000$, حداکثر یک ایستگاه بیشتر از 2 همسایه دارد.
- .4 (10 امتیاز) $k = 10^9$, $n \leq 8$
- .5 (61 امتیاز) $k = 10^9$

در زیرمسئله ۵ شما می‌توانید بخشی از نمره (partial score) را بگیرید. فرض کنید m بزرگ‌ترین لیلی باشد که توسط تابع `label` برای تمام سناریوها برگردانده شده است. نمره شما برای این زیرمسئله بر اساس جدول زیر محاسبه می‌شود:

بزرگ‌ترین لیل	نمره
$m \geq 10^9$	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

ارزیاب نمونه

ارزیاب نمونه ورودی را در قالب زیر می‌خواند

$$r:1$$

در ادامه r بلاک می‌آیند که هر کدام یک سناریو را توضیح می‌دهند. فرمت هر بلاک به صورت زیر است:

- خط $n \ k : 1$
- خط $u[i] \ v[i] : (0 \leq i \leq n - 2) \ 2 + i$
- خط $q : 1 + n$: تعداد دفعاتی که تابع `find_next_station` صدا زده می‌شود.
- خط $j : 0 \leq j \leq q - 1$: `z[j]` `y[j]` `w[j]` : اندیس ایستگاه‌هایی که در زامین فراخوانی تابع `find_next_station` استفاده می‌شوند. ایستگاه $[j]$ بسته را نگه داشته است، ایستگاه $[j]$ `y` مقصد بسته است، و ایستگاه $[j]$ `w`، ایستگاهی است که بسته باید به آن در این مرحله فرستاده شود.

ارزیاب نمونه خروجی را در قالب زیر چاپ می‌کند:

- خط $m : 1$
- r بلاک که هر کدام به ترتیب مربوط به یک سناریو در ورودی است. فرمت هر بلاک به صورت زیر است:
 - خط $j : 1 + q \leq j \leq q - 1$: `اندیس` ایستگاهی که لیل آن توسط زامین فراخوانی تابع `find_next_station` در این سناریو برگردانده شده است.

توجه کنید که هر اجرای ارزیاب نمونه هر دو تابع `label` و `find_next_station` را فرا می‌خواند.