2020 亚太区信息学奥林匹克

APIO 2020

时间：2020 年 8 月 15 日 09:00 ～ 14:00

题目名称	粉刷墙壁	交换城市	有趣的旅途
题目类型	传统型	传统型	交互型
目录	`paint`	`swap`	`fun`
可执行文件名	`paint`	`swap`	`fun`
输入文件名	`paint.in`	`swap.in`	`fun.in`
输出文件名	`paint.out`	`swap.out`	`fun.out`
每个测试点时限	1.5 秒	2.0 秒	2.0 秒
内存限制	512 MB	512 MB	512 MB
子任务数目	5	6	5
测试点是否等分	否	否	否

提交源程序文件名

对于 C++ 语言	`paint.cpp`	`swap.cpp`	`fun.cpp`

编译选项

对于 C++ 语言	`-O2 -std=c++11`

注意事项

1. 本题面中的题目顺序与 APIO 网站中题目顺序可能不一致，请实时提交时注意题目名称。
2. 题目最终的编译选项以提交时网站上的编译选项为准。

粉刷墙壁（paint）

【题目描述】

距离上一次 Pak Dengklek 在他的家中粉刷墙壁已经过了一段时间，所以他想重新粉刷一次。他家的墙壁由 N 段组成，它们从 0 到 $N-1$ 编号. 本题中我们假设存在 K 种不同的颜色，颜色用从 0 到 $K-1$ 的整数表示（例如, 红色用 0 表示, 蓝色用 1 表示，以此类推）。Pak Dengklek 希望用第 $C[i]$ 种颜色来粉刷第 i 段的墙壁。

为了粉刷墙壁, Pak Dengklek 雇用了一家有 M 个承包商的承包商公司，承包商从 0 到 $M-1$ 编号。对 Pak Dengklek 来说不幸的是，承包商只愿意粉刷他们自己喜欢的颜色。具体来说，第 j 个承包商喜欢 $A[j]$ 种颜色，并且只想用下列颜色来粉刷墙壁：第 $B[j][0]$ 种颜色，第 $B[j][1]$ 种颜色，...，或第 $B[j][A[j]-1]$ 种颜色。

Pak Dengklek 可以给承包商公司提出一些要求。在单个要求中，Pak Dengklek 将给出两个参数 x 和 y, 其中 $0 \leq x < M$, $0 \leq y \leq N-M$。承包商公司将会指派第 $((x+l) \bmod M)$ 个承包商粉刷第 $(y+l)$ 段墙壁，其中 $0 \leq l < M$。如果存在一个 l 使得第 $((x+l) \bmod M)$ 个承包商不喜欢第 $C[y+l]$ 种颜色，那么该要求将无效。

Pak Dengklek 需要为每个要求付费，因此他想知道为了使墙壁中每个段都能用自己预期的颜色粉刷，他至少要提出多少个要求，或是确认他的预期无法达到。每一段墙壁可以被粉刷多次，但必须保证每次粉刷的颜色都是 Pak Dengklek 所预期的。

【具体实现】

你必须实现 `minimumInstructions` 函数：

- `minimumInstructions(N, M, K, C, A, B)` - 该函数将被评测库恰好调用一次。
  - N: 一个整数表示墙壁的段数。
  - M: 一个整数表示承包商的数量。
  - K: 一个整数表示颜色的种数。
  - C: 一个长度为 N 的整数序列，表示每段墙壁预期的颜色。
  - A: 一个长度为 M 的整数序列，表示承包商喜欢的颜色数。
  - B: 一个长度为 M 的每个元素为序列的序列，表示承包商喜欢的具体颜色。
  - 该函数必须返回一个整数，表示 Pak Dengklek 为了让墙壁按预期粉刷所需要提出的最小要求数；若预期无法达到则返回 -1。

【样例】

在第一个样例中, $N=8$, $M=3$, $K=5$, $C=[3,3,1,3,4,4,2,2]$, $A=[3,2,2]$, $B=[[0,1,2],[2,3],[3,4]]$。Pak Dengklek 可以提出下列的要求。

1. $x = 1, y = 0$. 这是一个有效的要求，第一个承包商可以粉刷第零段墙壁，第二个承包商可以粉刷第一段墙壁，第零个承包商可以粉刷第二段墙壁。
2. $x = 0, y = 2$. 这是一个有效的要求，第零个承包商可以粉刷第二段墙壁，第一个承包商可以粉刷第三段墙壁，第二个承包商可以粉刷第四段墙壁。
3. $x = 2, y = 5$. 这是一个有效的要求，第二个承包商可以粉刷第五段墙壁，第零个承包商可以粉刷第六段墙壁，第一个承包商可以粉刷第七段墙壁。

容易看出 Pak Dengklek 不能用少于 3 个的要求来达到预期，因此 `minimumInstructions(8, 3, 5, [3, 3, 1, 3, 4, 4, 2, 2], [3, 2, 2], [[0, 1, 2], [2, 3], [3, 4]])` 应该返回 3。

在第二个样例中，$N = 5$, $M = 4$, $K = 4$, $C = [1, 0, 1, 2, 2]$, $A = [2, 1, 1, 1]$, $B = [[0, 1], [1], [2], [3]]$。由于第三个承包商只喜欢第 3 种颜色但没有任何一段墙壁能被该颜色粉刷，Pak Dengklek 无法给出任何有效指令。因此 `minimumInstructions(5, 4, 4, [1, 0, 1, 2, 2], [2, 1, 1, 1], [[0, 1], [1], [2], [3]])` 应该返回 -1.

【条件限制】

对于 $0 \leq k < K$, 令 $f(k)$ 表示喜欢第 k 种颜色的承包商数量。

- $1 \leq N \leq 100\,000$.
- $1 \leq M \leq \min(N, 50\,000)$.
- $1 \leq K \leq 100\,000$.
- $0 \leq C[i] < K$.
- $1 \leq A[j] \leq K$.
- $0 \leq B[j][0] < B[j][1] < \ldots < B[j][A[j] - 1] < K$.
- $\sum f(k)^2 \leq 400\,000$.

【子任务 1（12 分）】

- $f(k) \leq 1$.

【子任务 2（15 分）】

- $N \leq 500$.
- $M \leq \min(N, 200)$.
- $\sum f(k)^2 \leq 1\,000$.

【子任务 3（13 分）】

- $N \leq 500$.
- $M \leq \min(N, 200)$.

【子任务 4（23 分）】

- $N \leq 20\,000$.
- $M \leq \min(N, 2\,000)$.

【子任务 5（37 分）】

- 无附加限制。

【样例评测库】

样例评测库将读入以下格式的数据：

```
N M K
C[0] C[1] ... C[N-1]
A[0] B[0][0] B[0][1] ... B[0][A[0]-1]
A[1] B[1][0] B[1][1] ... B[1][A[1]-1]
.
.
.
A[M-1] B[M-1][0] B[M-1][1] ... B[M-1][A[M-1]-1]
```

样例评测库将输出函数 `minimumInstructions` 的返回值。

交换城市（swap）

【题目描述】

印度尼西亚有 N 个城市以及 M 条双向道路，城市从 0 到 $N-1$ 编号，道路从 0 到 $M-1$ 编号。每条道路连接着两个不同的城市，第 i 条道路连接第 $U[i]$ 个城市与第 $V[i]$ 个城市，汽车行驶这条道路将耗费 $W[i]$ 个单位汽油。通过这些道路，任意两个城市间能够互相到达。

接下来的 Q 天中, 每天会有一对城市希望建立政治关系。具体来说，第 j 天，第 $X[j]$ 个城市想要和第 $Y[j]$ 个城市建立政治关系。为此，第 $X[j]$ 个城市将会派一名代表坐汽车前往第 $Y[j]$ 个城市。同样地，第 $Y[j]$ 个城市也会派一名代表坐汽车前往第 $X[j]$ 个城市。

为了避免拥塞，两辆车不应在任何时间点碰面。更具体地，两辆车不能在同一个时间点出现在同一个城市。同样地，两辆车也不应该沿相反的方向同时行驶过同一条道路。另外，汽车行驶过一条道路时必须完整经过道路并到达道路另一端的城市（换句话说，汽车不允许在道路中间掉转方向）。但是，汽车可以多次到达一个城市或是多次经过一条道路。此外，汽车可以在任何时间在任何城市等候。

由于高燃料容量汽车的价格昂贵，两个城市都分别希望选择一条路线，使得两辆汽车所需的最大单位汽油容量最小。每个城市中都有加油站并且供油量是无限的，因此汽车所需的单位汽油容量实际上就是行驶过的道路中最大的单位汽油消耗量。

【具体实现】

你必须实现 `init` 和 `getMinimumFuelCapacity` 函数。

- `init(N, M, U, V, W)` - 该函数将在所有 `getMinimumFuelCapacity` 的调用前被评测库恰好调用一次。
  - N: 一个整数表示城市数。
  - M: 一个整数表示道路数。
  - U: 一个长为 M 的整数序列表示道路的第一个端点城市。
  - V: 一个长为 M 的整数序列表示道路的第二个端点城市。
  - W: 一个长为 M 的整数序列表示道路的汽油消耗。
- `getMinimumFuelCapacity(X, Y)` - 该函数将被评测库调用恰好 Q 次。
  - X: 一个整数表示第一个城市。
  - Y: 一个整数表示第二个城市。
  - 该函数必须返回一个整数，表示根据题目描述中的规则，两辆分别从第 X 个城市与第 Y 个城市出发要到达彼此城市的车，它们的单位汽油容量最大值的最小值。若无法满足题目规则则返回 -1。

【样例】

第一个样例中，$N = 5$，$M = 6$，$U = [0, 0, 1, 1, 1, 2]$，$V = [1, 2, 2, 3, 4, 3]$，$W = [4, 4, 1, 2, 10, 3]$，$Q = 3$，$X = [1, 2, 0]$，$Y = [2, 4, 1]$。如下图：

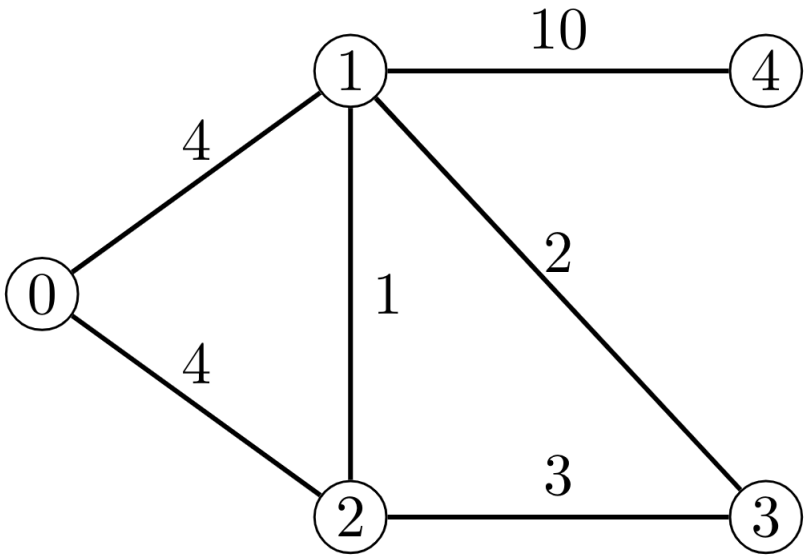

图 1: example1

评测库初始时将调用 `init(5, 6, [0, 0, 1, 1, 1, 2], [1, 2, 2, 3, 4, 3], [4, 4, 1, 2, 10, 3])`。之后，评测库将进行如下函数调用：

- `getMinimumFuelCapacity(1, 2)`。首先，从第一个城市出发的汽车可以行驶到第三个城市。接着，从第二个城市出发的汽车可以行驶到第一个城市，并且在第三个城市的汽车可以行驶到第二个城市。因此，最大的单位汽油容量为 3 （从第三个城市到第二个城市需要花费 3 个单位汽油）。没有其他更优的路线方案，因此该函数应该返回 3。
- `getMinimumFuelCapacity(2, 4)`。任何从第四个城市出发或要到达第四个城市的汽车都需要耗费 10 个单位汽油，因此该函数应该返回 10。
- `getMinimumFuelCapacity(0, 1)`。该函数应该返回 4。

第二个样例中，$N = 3$，$M = 2$，$U = [0, 0]$，$V = [1, 2]$，$W = [5, 5]$，$Q = 1$，$X = [1]$，$Y = [2]$。 如下图：

评测库初始时将调用 `init(3, 2, [0, 0], [1, 2], [5, 5])`. 之后，评测库将进行如下函数调用：

- `getMinimumFuelCapacity(1, 2)`。两辆车无法满足不在同一时间点碰面的要求，所以该函数应该返回 -1。

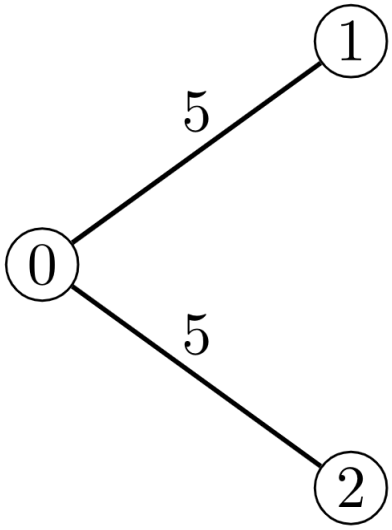

图 2: example2

【条件限制】

- $2 \leq N \leq 100\,000$.
- $N - 1 \leq M \leq 200\,000$.
- $0 \leq U[i] < V[i] < N$.
- 任意两个城市间至多存在一条道路直接相连。
- 任意两个城市经过道路可以互相到达。
- $1 \leq W[i] \leq 10^9$.
- $1 \leq Q \leq 200\,000$.
- $0 \leq X[j] < Y[j] < N$.

【子任务 1（6 分）】

- 每个城市至多是两条道路的一个端点。

【子任务 2（7 分）】

- $M = N - 1$.
- $U[i] = 0$.

【子任务 3（17 分）】

- $Q \leq 5$.
- $N \leq 1\,000$.

- $M \leq 2\,000$.

【子任务 4（20 分）】

- $Q \leq 5$.

【子任务 5（23 分）】

- $M = N - 1$.

【子任务 6（27 分）】

- 无附加限制。

【样例评测库】

样例评测库将读入以下格式的数据：

```
N M
U[0] V[0] W[0]
U[1] V[1] W[1]
.
.
.
U[M-1] V[M-1] W[M-1]
Q
X[0] Y[0]
X[1] Y[1]
.
.
.
X[Q-1] Y[Q-1]
```

对每个 `getMinimumFuelCapacity` 的调用，样例评测库会输出该函数的返回值。

有趣的旅途（fun）

【题目描述】

雅加达最大的主题公园中有 N 个景点，它们从 0 到 $N-1$ 编号。这些景点由 $N-1$ 条双向道路连接，任意两个景点间经由这些道路将存在唯一一条简单路径。道路从 0 到 $N-2$ 编号。第 i 条道路连接第 $A[i]$ 个景点与第 $B[i]$ 个景点，经过这条道路需要花费一个小时。为了避免拥塞，每个景点将至多与三条道路相连。

你想寻找一条游玩路线并使得每个景点都被参观一次。你认为从一个景点走到下一个景点时经过太多道路是十分无聊的。为了寻找一条有趣的路线，你打算安排景点的参观顺序，使得参观下一个景点所花费的时间不超过参观之前景点所花费的时间。换句话说，你想找到一个序列 $P[0], P[1], \ldots, P[N-1]$ 使其包含 0 到 $N-1$ 中的所有整数恰好一次，并且从第 $P[i]$ 个景点到达第 $P[i+1]$ 个景点所需的时间不超过从第 $P[i-1]$ 个景点到达第 $P[i]$ 个景点所需的时间，其中 $0 < i < N-1$。

你手上没有景点的完整地图，因此你必须向信息中心进行若干次询问才能找到一条有趣路线。你最多能进行 Q 次询问，每次询问需要提供两个参数 X 和 Y，其中 $0 \leq X, Y < N$。每次询问是以下任意一种：

- 从第 X 个景点到第 Y 个景点需要花费多少个小时。特别地，若 $X = Y$ 则回答将是 0。
- 有多少个景点 Z 满足，当你想从第 X 个景点到达第 Z 个景点时一定会经过第 Y 个景点。第 Y 个景点将会被计算在内，特别地，若 $X = Y$ 则回答将是 N。

【具体实现】

你必须实现 `createFunTour` 函数:

- `createFunTour(N, Q)` - 该函数将被评测库恰好调用一次。
  - N: 一个整数表示景点的数量。
  - Q: 一个整数表示询问次数的最大值。
  - 该函数可以调用以下两个交互函数：
    - `hoursRequired(X, Y)`
      - X: 一个整数表示第一个景点的编号。
      - Y: 一个整数表示第二个景点的编号。
      - 该函数将返回一个整数表示从第 X 个景点到第 Y 个景点需要花费的小时数。
      - 如果 X 或 Y 的值不在 0 到 $N-1$ 的范围内，该测试点将视为答案错误。
    - `attractionsBehind(X, Y)`
      - X: 一个整数表示第一个景点的编号。

- Y: 一个整数表示第二个景点的编号。
- 该函数将返回一个整数表示有多少个景点 Z 满足，当你想从第 X 个景点到达第 Z 个景点时一定会经过第 Y 个景点。
- 如果 X 或 Y 的值不在 0 到 $N-1$ 的范围内，该测试点将视为答案错误。

– 该函数必须返回一个长为 N 的整数序列，表示你找到的景点参观顺序。

【样例】

在下图的例子中 $N = 7$, $Q = 400\,000$, $A = [0, 0, 0, 1, 1, 2]$, $B = [1, 5, 6, 2, 4, 3]$。

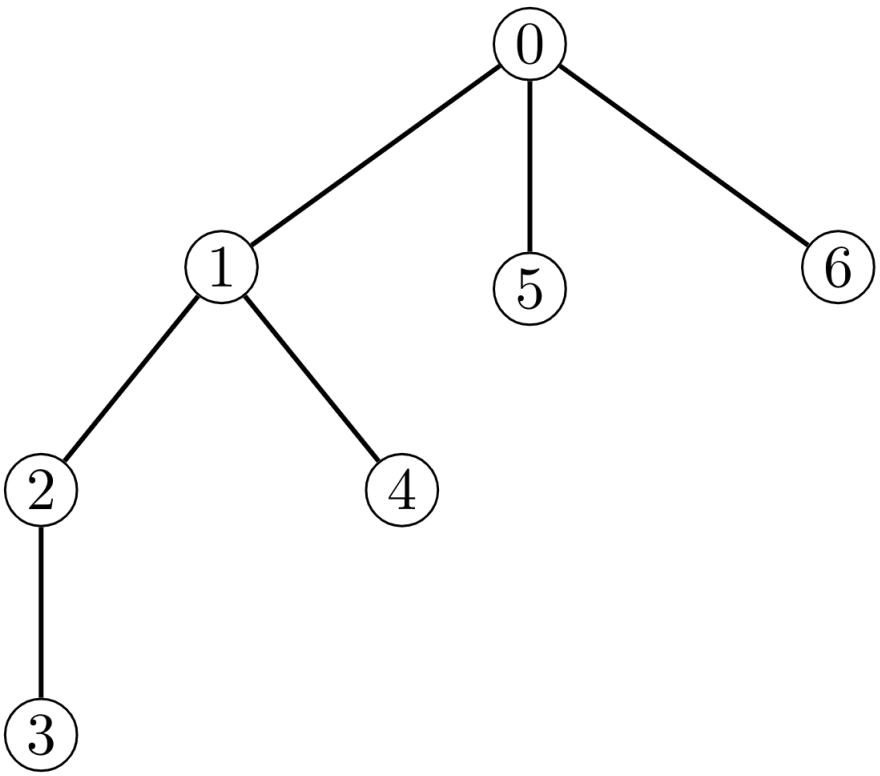

图 3: example1

评测库将调用 `createFunTour(7, 400000)`。

- 如果你询问 `hoursRequired(3, 5)`，函数将返回 4。
- 如果你询问 `hoursRequired(5, 4)`，函数将返回 3。
- 如果你询问 `attractionsBehind(5, 1)`，函数将返回 4。从第五个景点到第一、二、三、四个景点将一定会经过第一个景点。
- 如果你询问 `attractionsBehind(1, 5)`，函数将返回 1。

- 一个符合要求的返回序列为 $[3, 6, 4, 5, 2, 0, 1]$，到达下一个参观景点所需的时间按顺序分别为 $[4, 3, 3, 3, 2, 1]$。

【条件限制】

- $2 \leq N \leq 100\,000$.
- $Q = 400\,000$.
- 任意两个景点间可以通过双向道路互相到达。
- 每个景点至多连接着三条道路。

【子任务 1（10 分）】

- $N \leq 17$.

【子任务 2（16 分）】

- $N \leq 500$.

【子任务 3（21 分）】

- 对所有的 $1 \leq i < N$，有一条连接着第 i 个景点与第 $\left\lfloor \frac{i-1}{2} \right\rfloor$ 个景点的双向道路。

【子任务 4（19 分）】

- 存在至少一个景点 T 使得对于所有 $0 \leq i < N$, hoursRequired(T, i) < 30 并且存在一个整数区间 $[L[i], R[i]]$ $(0 \leq L[i] \leq i \leq R[i] < N)$ 满足下列条件：
  - 从第 T 个景点到达第 j 个景点必须经过第 i 个景点当且仅当 $L[i] \leq j \leq R[i]$。
  - 若 $L[i] < i$, 则恰有一个景点 X 满足：
    - $L[i] \leq X < i$.
    - 有一条连接第 i 个景点与第 X 个景点的道路。
  - 若 $i < R[i]$, 则恰有一个景点 Y 满足：
    - $i < Y \leq R[i]$.
    - 有一条连接第 i 个景点与第 Y 个景点的道路。

【子任务 5（34 分）】

- 无附加限制。

【样例评测库】

样例评测库将读入以下格式的数据：

```
N Q
A[0] B[0]
A[1] B[1]
.
.
.
A[N-2] B[N-2]
```

如果 createFunTour 正确返回了一个满足题意的序列，并且 hoursRequired 和 attractionsBehind 的调用次数总和不超过 Q,那么样例评测库将会输出 createFunTour 得到的序列。其他情况下样例评测库将会输出错误信息。