渡船（ferry）

【题目描述】

海上散布着 n 座岛屿，岛屿之间共有 m 条双向航线。第 i $(1 \le i \le m)$ 条航线连接岛屿 u_i 与 v_i，并且在这条航线上有一艘渡船往返运行。

每艘渡船有一个初始停泊位置。对于第 i $(1 \le i \le m)$ 条航线上的渡船，

- 若 $w_i = 0$，则渡船初始停泊在岛屿 u_i；
- 若 $w_i = 1$，则你可以任意决定渡船初始停泊在岛屿 u_i 或岛屿 v_i。

现在有 k 位旅人，第 j $(1 \le j \le k)$ 位旅人初始位于岛屿 s_j，需要前往岛屿 t_j。旅人只能通过渡船移动。具体地，每次移动会选择一位旅人 j $(1 \le j \le k)$ 与一条航线 i $(1 \le i \le m)$，满足旅人 j 与航线 i 上的渡船当前位于同一座岛屿，然后旅人将会登上渡船，与渡船一同移动到航线 i 另一端的岛屿。例如，若旅人 j 与航线 i 上的渡船均位于岛屿 u_i，则移动后旅人与渡船均会抵达岛屿 v_i，反之亦然。

你需要判断，是否可以通过一系列旅人的移动，使得所有旅人均能到达各自的目的地。若存在，你还需要给出一组具体的移动方案。

【实现细节】

选手不需要，也不应该实现 `main` 函数。

选手需要确保提交的程序包含头文件 `ferry.h`，即在程序开头加入以下代码：

```
#include "ferry.h"
```

选手需要在提交的程序源文件中实现以下函数：

```
void ferry(int n, int m, int k, std::vector<int> u,
   std::vector<int> v, std::vector<int> w, std::vector<int> s,
   std::vector<int> t);
```

- n, m, k 分别表示岛屿的数量、航线的数量与旅人的数量。
- 对于 $0 \le i < m$，u_i, v_i 分别表示第 $i+1$ 条航线连接的两座岛屿，w_i 表示该航线上的渡船的初始停泊位置是否确定，具体见【题目描述】。
- 对于 $0 \le j < k$，s_j, t_j 分别表示第 $j+1$ 位旅人初始位于的岛屿与目的地岛屿。
- 对于每个测试点，该函数会被交互库调用**恰好一次**。

选手可以通过调用以下函数报告是否可以通过一系列旅人的移动，使得所有旅人均能到达各自的目的地：

```
void report(bool o);
```

- 若 o 为 `true`，则表示可以通过一系列旅人的移动，使得所有旅人均能到达各自的目的地；若 o 为 `false`，则表示不可以。

- 选手需要保证交互库调用 `ferry` 时，**恰好调用了一次该函数**。

选手可以通过调用以下函数进行一次移动：

```
void move(int j, int i);
```

- j, i 分别表示选择的旅人与航线的编号，具体含义如【题目描述】中所示。选手需要保证 $1 \le j \le k$，$1 \le i \le m$，且旅人 j 与航线 i 上的渡船当前位于同一座岛屿。特别地，若航线 i 上的渡船的初始停泊位置还未确定，且旅人 j 位于航线 i 连接的其中一座岛屿，则航线 i 上的渡船的初始停泊位置直接被确定为旅人 j 所在的岛屿。
- 选手需要保证交互库每次调用 `ferry` 时，调用该函数的次数不超过 10^6，且所有该函数的调用**均在调用 `report` 函数之后，且 `report` 函数的返回值必须为 `true`**。

注意：在任何情况下，交互库运行所需时间均不会超过 0.1 秒，所用内存为固定大小，且均不超过 64 MiB。

【测试程序方式】

试题目录下的 `grader.cpp` 是交互库参考实现，最终测试时所用的交互库实现与该参考实现有所不同，因此选手的解法不应该依赖交互库实现。

选手可以在本题目录下使用如下命令编译得到可执行程序：

```
g++ grader.cpp ferry.cpp -o ferry -std=gnu++14 -O2 -pipe -static -s
```

对于编译得到的可执行程序：

- 可执行文件将从**标准输入**读入以下格式的数据：
  - 输入的第一行包含三个正整数 n, m, k，分别表示岛屿的数量、航线的数量与旅人的数量。
  - 输入的第 $i+1$ $(1 \le i \le m)$ 行包含三个非负整数 u_i, v_i, w_i，分别表示第 i 条航线连接的两座岛屿与该航线上的渡船的初始停泊位置是否确定，具体见【题目描述】。
  - 输入的第 $n+2$ 行包含 k 个正整数 $s_1, \ldots, s_k$，分别表示每位旅人初始位于的岛屿。
  - 输入的第 $n+3$ 行包含 k 个正整数 $t_1, \ldots, t_k$，分别表示每位旅人的目的地岛屿。
- 可执行文件将输出以下格式的数据至**标准输出**：
  - 输出的第一行包含一个字符串 `Yes` 或 `No`，表示是否可以通过一系列旅人的移动，使得所有旅人均能到达各自的目的地。
  - 若可以，则

* 输出的第二行包含一个非负整数 c，表示移动次数；
* 输出的第 $l+2$ $(1 \leq i \leq c)$ 行包含两个正整数 j, i，分别表示第 l 次移动选择的旅人与航线的编号。

【样例 1 输入】

```
4 5 2
1 2 1
3 1 0
3 2 0
2 4 1
4 3 0
1 4
3 2
```

【样例 1 输出】

```
Yes
4
2 5
2 3
1 1
1 3
```

【样例 1 解释】

- 初始时，旅人 1 位于岛屿 1，旅人 2 位于岛屿 4；
- 第一次移动后，旅人 2 与航线 5 上的渡船一同移动到岛屿 3；
- 第二次移动后，旅人 2 与航线 3 上的渡船一同移动到岛屿 2；
- 第三次移动时，航线 1 上的渡船的初始停泊位置被确定为岛屿 1，移动后旅人 1 与航线 1 上的渡船一同移动到岛屿 2；
- 第四次移动后，旅人 1 与航线 3 上的渡船一同移动到岛屿 3。

【附加文件说明】

在附加文件中：

1. `grader.cpp` 是提供的交互库参考实现。
2. `ferry.h` 是头文件，选手不用关心具体内容。

3. `template_ferry.cpp` 是提供的示例代码，选手可参考并实现自己的代码。

【子任务】

对于所有测试数据，均有：

- $2 \le n \le 2,000$，$1 \le m \le 10^4$，$1 \le k \le 10^2$；
- 对于所有 $1 \le i \le m$，均有 $1 \le u_i, v_i \le n$，$u_i \ne v_i$，$w_i \in \{0, 1\}$；
- 对于所有 $1 \le i \le k$，均有 $1 \le s_i, t_i \le n$。

子任务编号	分值	特殊性质
1	6	$k = 1$
2	7	$k = n$，且 $s_1, \ldots, s_n$ 与 $t_1, \ldots, t_n$ 均为 $1 \sim n$ 的排列
3	11	对于所有 $1 \le i \le m$，均有 $w_i = 0$
4	9	对于所有 $1 \le i \le m$，均有 $w_i = 1$
5	29	存在一个 $1 \sim k$ 的排列 p，满足对于所有 $1 \le i \le k$，均有 $s_i = t_{p_i}$
6	26	$n \le 100$，$m \le 200$，$k \le 60$
7	12	无

【评分方式】

注意：

- 选手不应当通过非法方式获取交互库的内部信息，如直接与标准输入、输出流进行交互。此类行为将被视为作弊；
- **最终的评测交互库与样例交互库的实现不同。**

本题首先会受到和传统题相同的限制，例如编译错误会导致整道题目得 0 分，运行时错误、超过时间限制、超过空间限制等会导致相应测试点得 0 分等。选手只能在程序中访问自己定义的变量以及交互库给出的变量，尝试访问其他地址空间将可能导致编译错误或运行错误。

每次调用 `ferry` 函数时，若 `report` 函数或 `move` 函数调用不合法，或 `move` 函数调用次数超过 10^6，则相应测试点得 0 分。

在上述条件基础上：

- 对于每个测试点，若 `report` 函数返回值正确，则可以获得 40% 的分数；在此基础上，若 `report` 返回值为 `false`，或 `ferry` 函数返回时所有旅人均到达对应的目的地岛屿，则可以获得满分。