星图（starmap）

【题目背景】

星图铺就的，未必是归途。
但有人循着它，便不算迷路。

【题目描述】

夜空中共有 n 颗星辰，编号为 $1 \sim n$。初始时，夜空中共有 m 条光轨，其中第 i $(0 \le i < m)$ 条光轨连接星辰 u_i 与 v_i。

古籍中记载了一种神秘的仪式，可以改变夜空中的光轨的状态。具体地，每次举行仪式时，需要选择**恰好** k 颗互不相同的星辰，然后改变这些星辰之间的光轨的状态。具体地，设选择星辰的编号分别为 $s_0, \ldots, s_{k-1}$，则对于所有 $0 \le i < j \le k-1$，若星辰 s_i 与星辰 s_j 之间此时存在光轨，则该光轨会被删除；若不存在，则会新增一条连接它们的光轨。由于仪式的耗材有限，这种仪式只能举行**至多** p $(p \ge n(n-1)/2)$ 次。

当存在更多的光轨时，夜空也会变得更加璀璨。你需要求出在举行至多 p 次仪式后，夜空中光轨数量的最大值，并尽可能地给出一种相应的举行仪式的方案。

【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 `starmap.h`，即在程序开头加入以下代码：

```
#include "starmap.h"
```

选手需要在提交的程序源文件 `starmap.cpp` 中实现以下两个函数：

```
void init(int c, int t);
```

- c, t 分别表示测试点编号与测试数据组数。$c = 0$ 表示该测试点为样例。
- 对于每个测试点，该函数会在**程序开始运行时**被交互库调用**恰好一次**。

```
void starmap(int n, int m, int k, int p, std::vector<int> u,
  std::vector<int> v);
```

- n, m, k, p 分别表示星辰数量、光轨数量、举行仪式时选择的星辰数量与举行仪式的次数限制。
- 对于 $0 \le i < m$，u_i, v_i 表示初始时的第 i 条光轨连接的两颗星辰。
- 对于每个测试点，该函数会被交互库调用**恰好** t **次**。

选手可以通过调用以下函数报告光轨数量的最大值：

```
void report(int c);
```

- c 表示光轨数量的最大值。
- 选手需要保证交互库调用 `starmap` 时，**恰好调用了一次该函数**。

选手可以通过调用以下函数举行一次仪式：

```
void invert(std::vector<int> s);
```

- $s_0, \ldots, s_{k-1}$ 表示选择的 k 颗星辰。选手需要保证 s 的长度为 k，且对于所有 $0 \le i \le k-1$，$1 \le s_i \le n$，且 $s_0, \ldots, s_{k-1}$ 互不相同。
- 选手需要保证交互库每次调用 `starmap` 时，调用该函数的次数不超过 p，且所有该函数的调用**均在调用 `report` 函数之后**。

注意：在任何情况下，最终测试时所用的交互库运行所需时间均不会超过 4.5 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 starmap.cpp -o starmap -std=gnu++14 -O2
   -static
```

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：
  - 输入的第一行包含两个非负整数 c, t，分别表示测试点编号和测试数据组数。
  - 接下来依次为每组测试数据，对于每组测试数据：
    - 第一行包含四个正整数 n, m, k, p，分别表示星辰数量、光轨数量、举行仪式时选择的星辰数量与举行仪式的次数限制。
    - 第 $i+2$ $(0 \le i < m)$ 行包含两个非负整数 u_i, v_i，表示初始时的第 i 条光轨连接的两颗星辰。
- 可执行文件将输出以下格式的数据至**标准输出**：
  - 对于每组测试数据，输出一行两个非负整数，分别表示光轨数量的最大值与举行所有仪式后光轨数量是否与最大值相等。

本题包含两个小问，正确回答第一个小问，即 `report` 函数报告的光轨数量的最大值正确，则可获得部分分数。具体评分规则请参见【评分方式】。

【样例 1 输入】

```
0 1
4 2 2 20
1 2
2 3
```

【样例 1 输出】

```
6 1
```

【样例 2 输入】

```
0 1
6 1 3 20
1 2
```

【样例 2 输出】

```
13 1
```

【样例 3】

见选手目录下的 ***starmap/starmap3.in*** 与 ***starmap/starmap3.ans***。
该样例满足测试点 3, 4 的约束条件。

【样例 4】

见选手目录下的 ***starmap/starmap4.in*** 与 ***starmap/starmap4.ans***。
该样例满足测试点 5 ∼ 7 的约束条件。

【样例 5】

见选手目录下的 ***starmap/starmap5.in*** 与 ***starmap/starmap5.ans***。
该样例满足测试点 8 ∼ 10 的约束条件。

【样例 6】

见选手目录下的 ***starmap/starmap6.in*** 与 ***starmap/starmap6.ans***。
该样例满足测试点 11 ∼ 13 的约束条件。

【样例 7】

见选手目录下的 ***starmap/starmap7.in*** 与 ***starmap/starmap7.ans***。
该样例满足测试点 $17 \sim 20$ 的约束条件。

【样例 8】

见选手目录下的 ***starmap/starmap8.in*** 与 ***starmap/starmap8.ans***。
该样例满足测试点 $21 \sim 25$ 的约束条件。

【下发文件说明】

在本试题目录下：

1. `grader.cpp` 是提供的交互库参考实现。
2. `starmap.h` 是头文件，选手不用关心具体内容。
3. `template_starmap.cpp` 是提供的示例代码，选手可参考并实现自己的代码。

选手注意对所有下发文件做好备份。最终评测时只测试本试题目录下的 `starmap.cpp`，对该程序以外文件的修改不会影响评测结果。

【数据范围】

设 N 为单个测试点内所有测试数据的 n 的和。对于所有测试数据，均有：

- $1 \le t \le 10$；
- $4 \le n \le 500$，$N \le 3,000$；
- $0 \le m \le n(n-1)/2$，$2 \le k \le n-2$，$n(n-1)/2 \le p \le 2 \times 10^5$；
- 对于所有 $0 \le i \le m-1$，均有 $1 \le u_i < v_i \le n$，且 $(u_0, v_0), \ldots, (u_{m-1}, v_{m-1})$ 互不相同。

测试点编号	$n \le$	k	$p =$
$1, 2$	8	$\le n-2$	500
$3, 4$	18		
$5 \sim 7$	500	$=3$	$n(n-1)/2$
$8 \sim 10$	70	$\le n-2$	
$11 \sim 13$	500	≤ 70	
$14 \sim 16$	300	$\le n-2$	$2n^2+5n$
$17 \sim 20$	400		n^2+10n
$21 \sim 25$	500		$n(n-1)/2$

【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 `starmap` 函数时，若 `report` 函数或 `invert` 函数调用不合法，或 `invert` 函数调用次数超过 p 次，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` 函数报告的光轨数量的最大值正确，则可以获得 25% 的分数；
- 在此基础上，若举行所有仪式后光轨数量与最大值相等，则可以获得满分。
- **注意：若报告的光轨数量的最大值正确，然而 `invert` 函数调用次数超过 p 次，则仍获得 0 分。**