题目名称	kanzenkankaku	retribution	medrcy
题目类型	传统型	传统型	传统型
输入文件名	kanzenkankaku.in	retribution.in	medrcy.in
输出文件名	kanzenkankaku.out	retribution.out	medrcy.out
时间限制	1s	3s	5s
空间限制	1024MB	1024MB	1024MB

kanzenkankaku

Taka 找到了一个长为 N 的字符串 $S = (S_1, S_2, \cdots, S_N)$，每个字符可以用 $1 \dots C$ 之间的一个整数来表示。从这个字符串第 i 个位置到第 j 个位置的字符串 $(S_i, S_{i+1}, \cdots, S_j)$ 称作子串 (i, j)。如果子串 (i, j) 翻转后和原来相等，即 $(S_i, S_{i+1}, \cdots, S_j) = (S_j, S_{j-1}, \cdots, S_i)$，则称子串 (i, j) 是回文的。
Taka 进行如下操作来制作一个回文串：

1. 首先，选择 S 的一个子串。不妨设选择的子串为 T；
2. 将子串 T 按照升序排序，得到 T'；
3. 在 S 中，用 T' 替换 T，得到 S'。形式化地：Taka 选择一个子串 (i, j)，将 $S_i, S_{i+1}, \cdots, S_j$ 按升序排序得到 $T'_i, T'_{i+1}, \cdots, T'_j$，最终得到
   $S' = (S_1, S_2, \cdots, S_{i-1}, T'_i, T'_{i+1}, \cdots, T'_j, S_{j+1}, \cdots, S_N)$；
4. 在这之后，寻找 S' 中的最长的回文子串。

Taka 进行如上操作后，想要得到最长的回文子串。

现在 Taka 将字符串 S 交给了你，请你输出 Taka 进行如上操作后能得到的最长回文子串的长度。

输入格式

第一行两个空格分隔的正整数 N 和 C，分别表示字符串的长度和字符集大小；
接下来 N 行，第 i 行一个正整数 S_i，表示字符串 S 中第 i 个位置的字符。

输出格式

输出一行一个正整数，表示 Taka 进行操作后能得到的最长回文子串的长度。

样例 1

样例输入 1

```
12 26
26
17
17
17
1
26
1
17
19
20
1
14
```

样例输出 1

```
8
```

样例解释

样例输入中，$N=12, C=26, S=(26,17,17,17,1,26,1,17,19,20,1,14)$。$Taka$ 可以选择子串 $(4,8)$，将其按照升序排列，得到 $S'=(26,17,17,1,1,17,17,26,19,20,1,14)$，这样子串 $(1,8)$ 就是回文了。这个回文长度为 8，是最长可能得到的回文子串。

样例 2

样例输入 2

```
4 3
1
2
3
2
```

样例输出 2

```
3
```

数据范围与提示

对于全部数据，$1 \leq N, C \leq 3000, 1 \leq S_i \leq C$。

本题有两个子任务。只有该任务下测试点全部通过才能得到该子任务的分数。

Subtask	附加限制	分数
1	$N, C \leq 50$	20
2	$N, C \leq 300$	30
3	$C \leq 2$	15
4	$N, C \leq 3000$	35

retribution

题目描述

给你一个 $n \times m$ 的表格，每个格子上标有一个字符 $c_{i,j} \in \{\mathtt{U},\mathtt{D},\mathtt{L},\mathtt{R}\}$，分别代表上、下、左、右。在一个格点可以移动到一个与它相邻的格点，但移动方向**不能**与当前所在格点上的字符所代表的方向相同且不能移出表格。

给出 q 次询问，每次询问给出两个格点 s,t，询问能否从格点 s 通过若干次移动到达格点 t。

输入格式

为避免输入数据过大，本题使用特殊的读入方式。

第一行四个正整数 $n,m,q,seed$，其中 $seed$ 为输入参数。
接下来 n 行，每行 m 个字符描述表格。

```
namespace Input{
    mt19937_64 R;
    inline void init(int seed){
        R=mt19937_64(seed);
    }
    inline int get(int l,int r){
        uniform_int_distribution<int> distribution(l,r);
        return distribution(R);
    }
}
using Input::init;
using Input::get;
```

你需要在你的代码中加入此段代码，初始时调用 `init(seed);`

接下来 q 次调用 `int a=get(1,n),b=get(1,m),x=get(1,n),y=get(1,m);`，a,b,x,y 分别描述 s,t 的位置 $(a,b),(x,y)$，表示一次询问。

输出格式

输出一行 q 个字符，第 i 个字符为 0 表示第 i 个询问中 s 无法到达 t，为 1 则表示第 i 个询问中 s 可以到达 t。

样例 #1

样例输入 #1

```
2 3 6 0
RLD
RLU
2 1 1 3
2 2 1 3
2 3 1 1
2 3 1 2
1 2 2 3
1 3 2 3
```

样例输出 #1

```
010111
```

样例 #2

样例输入 #2

```
3 3 5 0
DRU
ULU
DRD
1 1 1 3
1 3 1 1
3 1 2 1
2 3 2 1
1 1 1 1
```

样例输出 #2

```
01111
```


```
3 3 5 0
DRU
ULU
DRD
1 1 1 3
1 3 1 1
3 1 2 1
2 3 2 1
1 1 1 1
```


样例输出 #2

样例 #3

样例输入 #3

```
2 3 6 10000001
RLD
RLU
```

样例输出 #3

```
110000
```

提示

样例解释

为了便于阅读，对于样例 1 和样例 2，直接输入了询问而不使用特殊方式读入。

对于样例 1 第一次询问，格点 $(2,1)$ 无法移动出第 1 列。

对于样例 1 第四次询问，格点 $(2,3)$ 可以移动到 $(2,2)$ 从而移动到 $(1,2)$。

对于样例 1 第六次询问，格点 $(1,3)$ 可以依次移动到 $(1,2),(2,2),(2,3)$。

数据规模

本题采用捆绑测试。

Subtask	$n\le$	$m\le$	$q\le$	**分值**
1	100	100	10^3	10
2	3	500	2×10^5	20
3	300	300	2×10^5	20
4	1800	1800	10^6	50

对于 100% 的数据，$1\le n,m\le 1.8\times10^3$，$1\le q\le 10^6$，$c_{i,j}\in\{\texttt{U},\texttt{D},\texttt{L},\texttt{R}\}$，$1\le a,x\le n$，$1\le b,y\le m$，$0\le s<2^{31}$。

---

medrcy

题目描述

几个世纪以来，神奇的 Bitoland 一直是 n 个贤者和 m 条咒语的家园。根据古老的魔法法则，每条咒语恰好被 $n-2$ 个贤者知道。所有的贤者都知道每条咒语都被他们中的一些确定的人所知，但他们不知道到底有多少条咒语存在。每个贤者，对于他所知道的每一条咒语，都清楚地知道其他哪些贤者知道它。然而，贤者不知道存在多少他不知道的咒语。特别的，一个贤者可能不知道任何咒语——在这种情况下，他不知道是否存在咒语（但他仍然知道，如果存在咒语，则正好有 $n-2$ 个贤者会知道它们）。

贤者每天中午都会在 Stumegabyte 森林里聚会，但他们在那里不会互相交流，他们只是各自问候对方并进行冥想，晚上他们都回到自己的小屋。贤者除了在见面时看到对方之外，并没有其他任何方式的交流。他们这样做是因为他们害怕约束他们的古老传统，其中规定，如果一个贤者发现有他不知道的咒语，他必须在当天午夜神不知鬼不觉地离开这里，并且永远不能回到 Bitoland。

有一天，一个流浪者来到了 Bitoland。在观察了几天这些贤者之后，他决定去见他们，在那里他不明智地对所有贤者宣布：「我已经注意到，你们中至少有一个贤者知道至少一条咒语！」

流浪者将在 Bitoland 再停留 k 天（最多一个月），每天观察聚会情况，但不会再多说什么。在这段时间里，会不会有一天，一些贤者不会在聚会上出现?

我们假设贤者的推断是完美的，也就是说，如果他们中的任何一个人能够从流浪者宣布的内容和他们所掌握的关于咒语的信息中推断出什么，那么现实情况一定是这样的，并且他们会这么做。

输入格式

第一行一个正整数 t，表示测试点个数。

对于每个测试点，第一行包含三个整数 n,m,k，分别表示贤者人数，咒语个数和流浪者会观察会议的天数。贤者从 1 到 n 编号。

接下来 m 行，每行两个整数 a_i,b_i ，表示除了贤者 a_i 和 b_i 之外其他所有贤者都知道这条咒语。

输出格式

对于每组数据，如果接下来 k 天的每一天，所有贤者都会来参加聚会，则输出一行 -1。否则输出两行。第一行包含两个整数 d 和 c，其中 d 表示有贤者没来聚会的最早的一次，c 表示这一次没来聚会的贤者个数。第二行包含 c 个整数，表示没来聚会的贤者编号，按从小到大的顺序输出。

样例 #1

样例输入 #1

```
4
3 2 7
1 2
2 3
3 3 7
1 2
2 3
1 3
5 3 1
1 5
2 4
1 5
5 2 2
2 4
1 5
```

样例输出 #1

```
1 1
2
2 3
1 2 3
-1
2 4
1 2 4 5
```

数据规模与约定

对于所有数据，$3 \le n, \sum n \le 1000$，$1 \le m, \sum m \le 3000$，$1 \le k \le 30$，$1 \le a_i < b_i \le n$。

子任务编号	$\sum n \le$	$\sum m \le$	$k \le$	分值
1	1000	3000	2	15
2	200	3000	3	10
3	20	3000	30	35
4	40	3000	30	10
5	1000	3000	30	30