实外 CCF CSP2023-S模拟赛(一)

（提高组）

时间：2023年 7 月 7 日 8:30 ~ 12:00

题目引用：NOIP2016、NOIP2018

题目名称	蚯蚓	填数游戏	愤怒的小鸟	保卫王国
题目类型	传统型	传统型	传统型	传统型
目录	earthworm	game	angrybirds	defense
可执行文件名	earthworm	game	angrybirds	defense
输入文件名	earthworm.in	game.in	angrybirds.in	defense.in
输出文件名	earthworm.out	game.out	angrybirds.out	defense.out
每个测试点时限	1.0 秒	1.0 秒	2.0 秒	2.0 秒
内存限制	512 MB	512 MB	512MB	512MB
子任务数目	20	20	20	25
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	earthworm.cpp	game.cpp	angrybirds.cpp	defense.cpp
对于 C 语言	earthworm.c	game.c	angrybirds.c	defense.c
对于 Pascal 语言	earthworm.pas	game.pas	angrybirds.pas	defense.pas

编译选项

对于 C++ 语言	-O2
对于 C 语言	-O2
对于 Pascal 语言	-O2

注意事项（请仔细阅读）

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. C/C++ 中函数 main() 的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
3. 提交的程序代码文件的放置位置请参考各省的具体要求。
4. 因违反以上三点而出现的错误或问题，申述时一律不予受理。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 程序可使用的栈空间内存限制与题目的内存限制一致。
7. 全国统一评测时采用的机器配置为：Inter(R) Core(TM) i7-8700K CPU @3.70GHz，内存 32GB。上述时限以此配置为准。
8. 只提供 Linux 格式附加样例文件。
9. 评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以此为准。

蚯蚓（earthworm）

【问题描述】

本题中，我们将用符号 $\lfloor c \rfloor$ 表示对 c 向下取整，例如：$\lfloor 3.0 \rfloor = \lfloor 3.1 \rfloor = \lfloor 3.9 \rfloor = 3$。

蛐蛐国最近蚯蚓成灾了！隔壁跳蚤国的跳蚤也拿蚯蚓们没办法，蛐蛐国王只好去请神刀手来帮他们消灭蚯蚓。

蛐蛐国里现在共有 n 只蚯蚓（n 为正整数）。每只蚯蚓拥有长度，我们设第 i 只蚯蚓的长度为 a_i ($i = 1, 2, \ldots, n$)，并保证所有的长度都是非负整数（即：可能存在长度为 0 的蚯蚓）。

每一秒，神刀手会在所有的蚯蚓中，准确地找到最长的那一只（如有多个则任选一个）将其切成两半。神刀手切开蚯蚓的位置由常数 p（是满足 $0 < p < 1$ 的有理数）决定，设这只蚯蚓长度为 x，神刀手会将其切成两只长度分别为 $\lfloor px \rfloor$ 和 $x - \lfloor px \rfloor$ 的蚯蚓。特殊地，如果这两个数的其中一个等于 0，则这个长度为 0 的蚯蚓也会被保留。此外，除了刚刚产生的两只新蚯蚓，其余蚯蚓的长度都会增加 q（是一个非负整常数）。

蛐蛐国王知道这样不是长久之计，因为蚯蚓不仅会越来越多，还会越来越长。蛐蛐国王决定求助于一位有着洪荒之力的神秘人物，但是救兵还需要 m 秒才能到来……（m 为非负整数）

蛐蛐国王希望知道这 m 秒内的战况。具体来说，他希望知道：

- m 秒内，每一秒被切断的蚯蚓被切断前的长度（有 m 个数）；
- m 秒后，所有蚯蚓的长度（有 $n + m$ 个数）。

蛐蛐国王当然知道怎么做啦！但是他想考考你……

【输入格式】

从文件***earthworm.in*** 中读入数据。

第一行包含六个整数 n, m, q, u, v, t，其中：n, m, q 的意义见【问题描述】；u, v, t 均为正整数；你需要自己计算 $p = u/v$（保证 $0 < u < v$）；t 是输出参数，其含义将会在【输出格式】中解释。

第二行包含 n 个非负整数，为 $a_1, a_2, \ldots, a_n$，即初始时 n 只蚯蚓的长度。

同一行中相邻的两个数之间，恰好用一个空格隔开。

保证 $1 \le n \le 10^5$，$0 \le m \le 7 \times 10^6$，$0 < u < v \le 10^9$，$0 \le q \le 200$，$1 \le t \le 71$，$0 \le a_i \le 10^8$。

【输出格式】

输出到文件***earthworm.out*** 中。

第一行输出 $\left\lfloor \frac{m}{t} \right\rfloor$ 个整数，按时间顺序，依次输出第 t 秒，第 $2t$ 秒，第 $3t$ 秒，……被切断蚯蚓（在被切断前）的长度。

第二行输出 $\left\lfloor \frac{n+m}{t} \right\rfloor$ 个整数，输出 m 秒后蚯蚓的长度；需要按从大到小的顺序，依次输出排名第 t，第 $2t$，第 $3t$，……的长度。

同一行中相邻的两个数之间，恰好用一个空格隔开。即使某一行没有任何数需要输出，你也应输出一个空行。

请阅读样例来更好地理解这个格式。

【样例1输入】

```
3 7 1 1 3 1
3 3 2
```

【样例1输出】

```
3 4 4 4 5 5 6
6 6 6 5 5 4 4 3 2 2
```

【样例1说明】

在神刀手到来前：3只蚯蚓的长度为3,3,2。

1秒后：一只长度为3的蚯蚓被切成了两只长度分别为1和2的蚯蚓，其余蚯蚓的长度增加了1。最终4只蚯蚓的长度分别为(1,2),4,3。括号表示这个位置刚刚有一只蚯蚓被切断。

2秒后：一只长度为4的蚯蚓被切成了1和3。5只蚯蚓的长度分别为：2,3,(1,3),4。

3秒后：一只长度为4的蚯蚓被切断。6只蚯蚓的长度分别为：3,4,2,4,(1,3)。

4秒后：一只长度为4的蚯蚓被切断。7只蚯蚓的长度分别为：4,(1,3),3,5,2,4。

5秒后：一只长度为5的蚯蚓被切断。8只蚯蚓的长度分别为：5,2,4,4,(1,4),3,5。

6秒后：一只长度为5的蚯蚓被切断。9只蚯蚓的长度分别为：(1,4),3,5,5,2,5,4,6。

7秒后：一只长度为6的蚯蚓被切断。10只蚯蚓的长度分别为：2,5,4,6,6,3,6,5,(2,4)。

所以，7秒内被切断的蚯蚓的长度依次为3,4,4,4,5,5,6。7秒后，所有蚯蚓长度从大到小排序为6,6,6,5,5,4,4,3,2,2。

【样例2输入】

```
3 7 1 1 3 2
3 3 2
```

【样例2输出】

```
4 4 5
6 5 4 3 2
```

【样例2说明】

这个数据中只有 $t = 2$ 与上个数据不同。只需在每行都改为每两个数输出一个数即可。

虽然第一行最后有一个6没有被输出，但是第二行仍然要重新从第二个数再开始输出。

【样例3输入】

```
3 7 1 1 3 9
3 3 2
```

【样例3输出】

```
2
```

【样例3说明】

这个数据中只有 $t = 9$ 与上个数据不同。

注意第一行没有数要输出，但也要输出一个空行。

【子任务】

- 测试点1 ~ 3 满足 $m = 0$。
- 测试点4 ~ 7 满足 $n, m \le 1,000$。
- 测试点8 ~ 14 满足 $q = 0$，其中测试点 8 ~ 9 还满足 $m \le 10^5$。
- 测试点15 ~ 18 满足 $m \le 3 \times 10^5$。
- 测试点19 ~ 20 没有特殊的约定，参见原始的数据范围。
- 测试点1 ~ 12，15 ~ 16 还满足 $v \le 2$，这意味着 u, v 的唯一可能的取值是 $u = 1, v = 2$，即 $p = 0.5$。这可能会对解决问题有特殊的帮助。

每个测试点的详细数据范围见下表。

测试点	n	m	t	a_i	v	q
1	$=1$	$=0$	$=1$	$\leq 10^6$	≤ 2	$=0$
2	$=10^3$					
3	$=10^5$					
4	$=1$	$=10^3$				
5	$=10^3$					
6	$=1$					≤ 200
7	$=10^3$					
8	$=5\times10^4$	$=5\times10^4$				$=0$
9	$=10^5$	$=10^5$	$=2$			
10		$=2\times10^6$	$=21$			
11		$=2.5\times10^6$	$=26$			
12		$=3.5\times10^6$	$=36$	$\leq 10^7$		
13		$=5\times10^6$	$=51$		$\leq 10^9$	
14		$=7\times10^6$	$=71$	$\leq 10^8$		
15	$=5\times10^4$	$=5\times10^4$	$=1$		≤ 2	≤ 200
16		$=1.5\times10^5$	$=2$			
17	$=10^5$	$=10^5$	$=3$		$\leq 10^9$	
18		$=3\times10^5$	$=4$			
19		$=3.5\times10^6$	$=36$			
20		$=7\times10^6$	$=71$			

填数游戏（game）

【问题描述】

小 D 特别喜欢玩游戏。这一天，他在玩一款填数游戏。

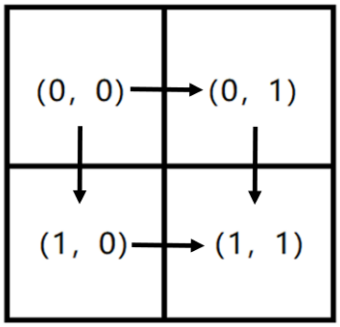
这个填数游戏的棋盘是一个$n \times m$的矩形表格。玩家需要在表格的每个格子中填入一个数字（数字 0 或者数字 1），填数时需要满足一些限制。

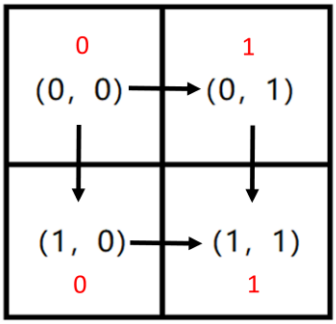
下面我们来具体描述这些限制。

为了方便描述，我们先给出一些定义：

- 我们用每个格子的行列坐标来表示一个格子，即（行坐标，列坐标）。（注意：行列坐标均从 0 开始编号）
- 合法路径 P：一条路径是合法的当且仅当：
  1. 这条路径从矩形表格的左上角的格子$(0,0)$出发，到矩形的右下角格子$(n-1, m-1)$结束；
  2. 在这条路径中，每次只能从当前的格子移动到右边与它相邻的格子，或者从当前格子移动到下面与它相邻的格子。

例如：在下面这个矩形中，只有两条路径是合法的，它们分别是P_1：$(0,0) \to (0,1) \to (1,1)$和$P_2$：$(0,0) \to (1,0) \to (1,1)$。





对于一条合法的路径 P，我们可以用一个字符串$w(P)$来表示，该字符串的长度为$n+m-2$，其中只包含字符“R”或者字符“D”，第 i 个字符记录了路径 P 中第 i 步的移动方法，“R”表示移动到当前格子右边与它相邻的格子，“D”表示移动到当前格子下面与它相邻的格子。例如，上图中对于路径P_1，有$w(P_1) = \text{"RD"}$；而对于另一条路径P_2，有$w(P_2) = \text{"DR"}$。

同时，将每条合法路径 P 经过的每个格子上填入的数字依次连接后，会得到一个长度为$n+m-1$的 01 字符串，记为 $s(P)$。例如，如果我们在格子$(0,0)$和$(1,0)$上填入数字 0，在格子$(0,1)$和$(1,1)$上填入数字 1（见上图红色数字）。那么对于路径P_1，我们可以得到$s(P_1) = \text{"011"}$,对于路径P_2，有$s(P_2) = \text{"001"}$。

游戏要求小 D 找到一种填数字 0、1 的方法，使得对于两条路径P_1，P_2，如果$w(P_1) > w(P_2)$，那么必须$s(P_1) \le s(P_2)$。我们说字符串 a 比字符串 b 小，当且仅当字符串 a 的字典序小于字符串 b 的字典序，字典序的定义详见第一题。但是仅仅是找一种方法无法满足小 D 的好奇心，小 D 更想知道这个游戏有多少种玩法，也就是说，有多少种填数字的方法满足游戏的要求？

小 D 能力有限，希望你帮助他解决这个问题，即有多少种填 0、1 的方法能满足题目要求。由于答案可能很大，你需要输出答案对10^9+7取模的结果。

【输入格式】

输入文件名为 game.in。

输入文件共一行，包含两个正整数 n、m，由一个空格分隔，表示矩形的大小。其中 n 表示矩形表格的行数，m 表示矩形表格的列数。

【输出格式】

输出文件名为 game.out。

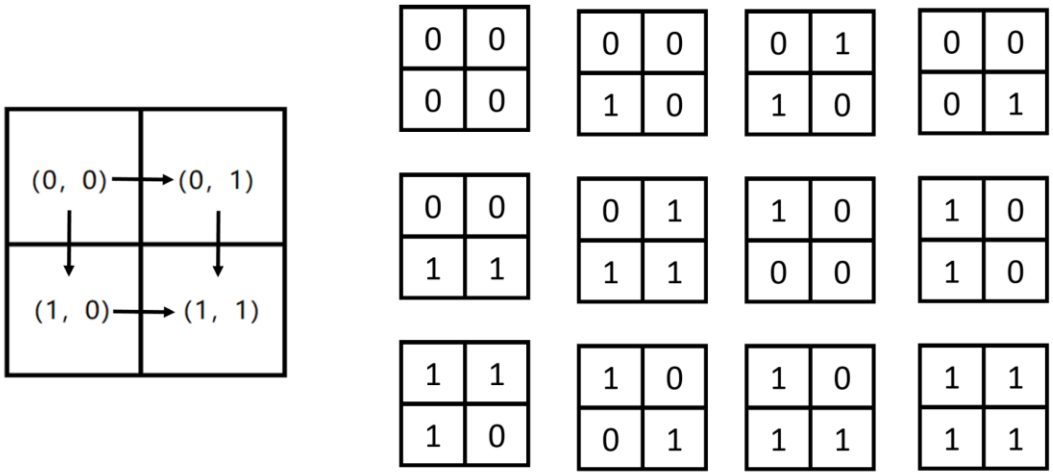
输出共一行，包含一个正整数，表示有多少种填 0、1 的方法能满足游戏的要求。注意：输出答案对 10^9+7 取模的结果。

【输入输出样例 1】

game.in	**game.out**
2 2	12

见选手目录下的 game/game1.in 和 game/game1.ans。

【样例解释】



对于2×2棋盘，有上图所示的 12 种填数方法满足要求。

【输入输出样例 2】

game.in	**game.out**
3 3	112

见选手目录下的 game/game2.in 和 game/game2.ans。。

【输入输出样例 3】

game.in	**game.out**
5 5	7136

见选手目录下的 game/game3.in 和 game/game3.ans。

【数据规模与约定】

测试点编号	n ≤	m ≤
1~4	3	3
5~10	2	1000000
11~13	3	1000000
14~16	8	8
17~20	8	1000000

愤怒的小鸟（angrybirds）

【问题描述】

Kiana最近沉迷于一款神奇的游戏无法自拔。

简单来说，这款游戏是在一个平面上进行的。

有一架弹弓位于 $(0,0)$ 处，每次Kiana可以用它向第一象限发射一只红色的小鸟，小鸟们的飞行轨迹均为形如 $y=ax^2+bx$ 的曲线，其中 a,b 是Kiana指定的参数，且必须满足 $a<0$。

当小鸟落回地面（即 x 轴）时，它就会瞬间消失。

在游戏的某个关卡里，平面的第一象限中有 n 只绿色的小猪，其中第 i 只小猪所在的坐标为 (x_i,y_i)。

如果某只小鸟的飞行轨迹经过了 (x_i,y_i)，那么第 i 只小猪就会被消灭掉，同时小鸟将会沿着原先的轨迹继续飞行；

如果一只小鸟的飞行轨迹没有经过 (x_i,y_i)，那么这只小鸟飞行的全过程就不会对第 i 只小猪产生任何影响。

例如，若两只小猪分别位于 $(1,3)$ 和 $(3,3)$，Kiana可以选择发射一只飞行轨迹为 $y=-x^2+4x$ 的小鸟，这样两只小猪就会被这只小鸟一起消灭。

而这个游戏的目的，就是通过发射小鸟消灭所有的小猪。

这款神奇游戏的每个关卡对Kiana来说都很难，所以Kiana还输入了一些神秘的指令，使得自己能更轻松地完成这个游戏。这些指令将在【输入格式】中详述。

假设这款游戏一共有 T 个关卡，现在Kiana想知道，对于每一个关卡，至少需要发射多少只小鸟才能消灭所有的小猪。由于她不会算，所以希望由你告诉她。

【输入格式】

从文件***angrybirds.in*** 中读入数据。

第一行包含一个正整数 T，表示游戏的关卡总数。

下面依次输入这 T 个关卡的信息。每个关卡第一行包含两个非负整数 n,m，分别表示该关卡中的小猪数量和Kiana输入的神秘指令类型。接下来的 n 行中，第 i 行包含两个正实数 x_i,y_i，表示第 i 只小猪坐标为 (x_i,y_i)。数据保证同一个关卡中不存在两只坐标完全相同的小猪。

如果 $m=0$，表示Kiana输入了一个没有任何作用的指令。

如果 $m=1$，则这个关卡将会满足：至多用 $\lceil n/3+1\rceil$ 只小鸟即可消灭所有小猪。

如果 $m=2$，则这个关卡将会满足：一定存在一种最优解，其中有一只小鸟消灭了至少 $\lfloor n/3\rfloor$ 只小猪。

保证 $1\le n\le 18$，$0\le m\le 2$，$0<x_i,y_i<10$，输入中的实数均保留到小数点后两位。

上文中，符号 $\lceil c \rceil$ 和 $\lfloor c \rfloor$ 分别表示对 c 向上取整和向下取整，例如：$\lceil 2.1 \rceil = \lceil 2.9 \rceil = \lceil 3.0 \rceil = \lfloor 3.0 \rfloor = \lfloor 3.1 \rfloor = \lfloor 3.9 \rfloor = 3$ 。

【输出格式】

输出到文件***angrybirds.out*** 中。

对每个关卡依次输出一行答案。

输出的每一行包含一个正整数，表示相应的关卡中，消灭所有小猪最少需要的小鸟数量。

【样例1输入】

```
2
2 0
1.00 3.00
3.00 3.00
5 2
1.00 5.00
2.00 8.00
3.00 9.00
4.00 8.00
5.00 5.00
```

【样例1输出】

```
1
1
```

【样例1说明】

这组数据中一共有两个关卡。

第一个关卡与【问题描述】中的情形相同， 2 只小猪分别位于 $(1.00, 3.00)$ 和 $(3.00, 3.00)$ ，只需发射一只飞行轨迹为 $y = -x^2 + 4x$ 的小鸟即可消灭它们。

第二个关卡中有 5 只小猪，但经过观察我们可以发现它们的坐标都在抛物线 $y = -x^2 + 6x$ 上，故Kiana只需要发射一只小鸟即可消灭所有小猪。

【样例2输入】

```
3
2 0
1.41 2.00
1.73 3.00
3 0
1.11 1.41
2.34 1.79
2.98 1.49
5 0
2.72 2.72
2.72 3.14
3.14 2.72
3.14 3.14
5.00 5.00
```

【样例2输出】

```
2
2
3
```

【样例3输入】

```
1
10 0
7.16 6.28
2.02 0.38
8.33 7.78
7.68 2.09
7.46 7.86
5.77 7.44
8.24 6.72
4.42 5.11
5.42 7.79
8.15 4.99
```

【样例3输出】

```
6
```

【子任务】

数据的一些特殊规定如下表：

测试点编号	n	m	T
1	≤ 2	= 0	≤ 10
2			≤ 30
3	≤ 3		≤ 10
4			≤ 30
5	≤ 4		≤ 10
6			≤ 30
7	≤ 5		≤ 10
8	≤ 6		
9	≤ 7		
10	≤ 8		
11	≤ 9		≤ 30
12	≤ 10		
13	≤ 12	= 1	
14		= 2	
15	≤ 15	= 0	≤ 15
16		= 1	
17		= 2	
18	≤ 18	= 0	≤ 5
19		= 1	
20		= 2	

保卫王国(defense)

【问题描述】

Z 国有n座城市，$n-1$条双向道路，每条双向道路连接两座城市，且任意两座城市都能通过若干条道路相互到达。

Z 国的国防部长小 Z 要在城市中驻扎军队。驻扎军队需要满足如下几个条件：

- 一座城市可以驻扎一支军队，也可以不驻扎军队。
- 由道路直接连接的两座城市中至少要有一座城市驻扎军队。
- 在城市里驻扎军队会产生花费，在编号为i的城市中驻扎军队的花费是p_i。

小 Z 很快就规划出了一种驻扎军队的方案，使总花费最小。但是国王又给小 Z 提出了m个要求，每个要求规定了其中两座城市是否驻扎军队。小 Z 需要针对每个要求逐一给出回答。具体而言，如果国王提出的第j个要求能够满足上述驻扎条件（不需要考虑第 j 个要求之外的其它要求），则需要给出在此要求前提下驻扎军队的最小开销。如果国王提出的第j个要求无法满足，则需要输出-1 ($1 \le j \le m$)。现在请你来帮助小 Z。

【输入格式】

输入文件名为 defense.in。

第 1 行包含两个正整数n,m和一个字符串$type$，分别表示城市数、要求数和数据类型。$type$是一个由大写字母 A，B 或 C 和一个数字 1，2，3 组成的字符串。它可以帮助你获得部分分。你可能不需要用到这个参数。这个参数的含义在【数据规模与约定】中有具体的描述。

第2行n个整数p_i，表示编号i的城市中驻扎军队的花费。

接下来$n-1$行，每行两个正整数u,v，表示有一条u到v的双向道路。

接下来m行，第j行四个整数$a,x,b,y(a \neq b)$，表示第j个要求是在城市a驻扎x支军队，在城市b驻扎y支军队。其中，x 、 y 的取值只有 0 或 1：若 x 为 0，表示城市 a 不得驻扎军队，若 x 为 1，表示城市 a 必须驻扎军队；若 y 为 0，表示城市 b 不得驻扎军队，若 y 为 1，表示城市 b 必须驻扎军队。

输入文件中每一行相邻的两个数据之间均用一个空格分隔。

【输出格式】

输出文件名为 defense.out。

输出共m行，每行包含 1 个整数，第j行表示在满足国王第j个要求时的最小开销，如果无法满足国王的第j个要求，则该行输出-1。

【输入输出样例 1】

defense.in	defense.out
5 3 C3 2 4 1 3 9 1 5 5 2 5 3 3 4 1 0 3 0 2 1 3 1 1 0 5 0	12 7 -1

见选手目录下的 defense/defense1.in 和 defense/defense1.ans。

【样例解释】

对于第一个要求，在 4 号和 5 号城市驻扎军队时开销最小。

对于第二个要求，在 1 号、2 号、3 号城市驻扎军队时开销最小。

第三个要求是无法满足的，因为在 1 号、5 号城市都不驻扎军队就意味着由道路直接连接的两座城市中都没有驻扎军队。

【输入输出样例 2】

见选手目录下的 defense/defense2.in 和 defense/defense2.ans。

【数据规模与约定】

对于 100%的数据，$n, m \leq 300000, 1 \leq p_i \leq 100000$。

测试点编号	type	$n=$	$m=$
1~2	A3	10	10
3~4	C3		
5~6	A3	100	100
7	C3		
8~9	A3	2000	2000
10~11	C3		
12~13	A1	100000	100000
14~16	A2		
17	A3		
18~19	B1		
20~21	C1		
22	C2		
23~25	C3		

数据类型的含义：

A：城市i与城市$i+1$直接相连。

B：任意城市与城市 1 的距离不超过 100（距离定义为最短路径上边的数量），即如果这棵树以 1 号城市为根，深度不超过 100。

C：在树的形态上无特殊约束。

1：询问时保证$a=1, x=1$，即要求在城市 1 驻军。对b, y没有限制。

2：询问时保证a, b是相邻的（由一条道路直接连通）

3：在询问上无特殊约束。