第 22 届全国青少年信息学奥林匹克联赛

CCF-NOIP-2016

提高组（复赛） 第一试

竞赛时间：2016 年 11 月 19 日 8:30 ~ 12:00

题目名称	玩具谜题	天天爱跑步	换教室
题目类型	传统型	传统型	传统型
目录	toy	running	classroom
可执行文件名	toy	running	classroom
输入文件名	toy.in	running.in	classroom.in
输出文件名	toy.out	running.out	classroom.out
每个测试点时限	1.0 秒	2.0 秒	1.0 秒
内存限制	512 MB	512 MB	512 MB
测试点数目	20	20	25
每个测试点分值	5	5	4

提交源程序文件名

对于 C++ 语言	toy.cpp	running.cpp	classroom.cpp
对于 C 语言	toy.c	running.c	classroom.c
对于 Pascal 语言	toy.pas	running.pas	classroom.pas

编译选项

对于 C++ 语言	-lm	-lm	-lm
对于 C 语言	-lm	-lm	-lm
对于 Pascal 语言			

注意事项：

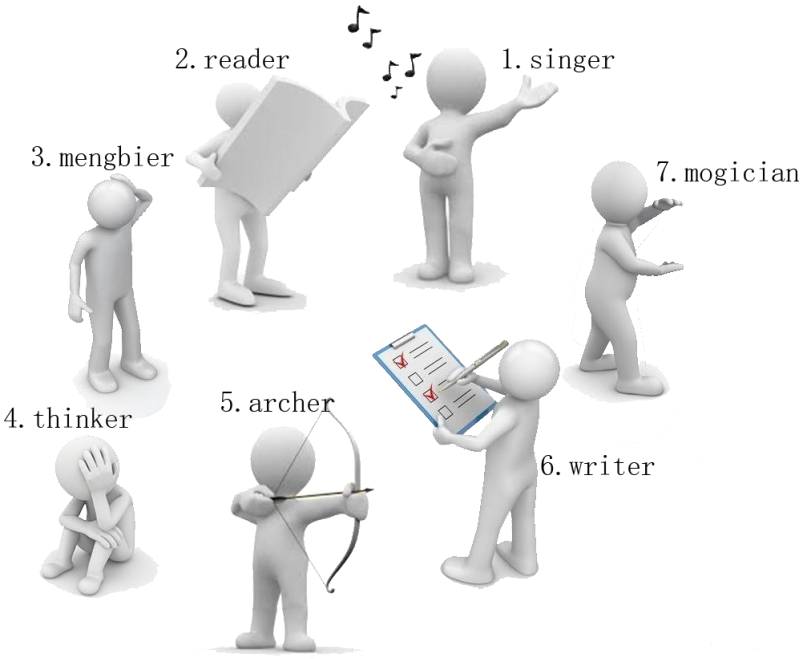
1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. 除非特殊说明，结果比较方式均为忽略行末空格及文末回车的全文比较。
3. C/C++中函数 main()的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
4. 全国统一评测时采用的机器配置为：CPU AMD Athlon(tm) II x2 240 processor，2.8GHz，内存 4G，上述时限以此配置为准。
5. 只提供 Linux 格式附加样例文件。
6. 评测在 NOI Linux 下进行。
7. 编译时不打开任何优化选项。

玩具谜题（toy）

【问题描述】

小南有一套可爱的玩具小人，它们各有不同的职业。

有一天，这些玩具小人把小南的眼镜藏了起来。 小南发现玩具小人们围成了一个圈，它们有的面朝圈内，有的面朝圈外。 如下图：



这时 singer 告诉小南一个谜题：“眼镜藏在我左数第 3 个玩具小人的右数第 1 个玩具小人的左数第 2 个玩具小人那里。”

小南发现，这个谜题中玩具小人的朝向非常关键，因为朝内和朝外的玩具小人的左右方向是相反的：面朝圈内的玩具小人，它的左边是顺时针方向，右边是逆时针方向；而面向圈外的玩具小人，它的左边是逆时针方向，右边是顺时针方向。

小南一边艰难地辨认着玩具小人，一边数着：

“singer 朝内，左数第 3 个是 archer。

“archer 朝外，右数第 1 个是 thinker。

“thinker 朝外，左数第 2 个是 writer。 “所以

眼镜藏在 writer 这里！”

虽然成功找回了眼镜，但小南并没有放心。 如果下次有更多的玩具小人藏他的眼　镜，或是谜题的长度更长，他可能就无法找到眼镜了。 所以小南希望你写程序帮他解　决类似的谜题。　这样的谜题具体可以描述为：

有 n 个玩具小人围成一圈，已知它们的职业和朝向。现在第 1 个玩具小人告诉小南一个包含 m 条指令的谜题，其中第 i 条指令形如“左数/右数第 s_i 个玩具小人”。你需要输出依次数完这些指令后，到达的玩具小人的职业。

【输入格式】

从文件 ***toy.in*** 中读入数据。

输入的第一行包含两个正整数 n,m，表示玩具小人的个数和指令的条数。

接下来 n 行，每行包含一个整数和一个字符串，以逆时针为顺序给出每个玩具小人的朝向和职业。其中 0 表示朝向圈内，1 表示朝向圈外。保证不会出现其他的数。字符串长度不超过 10 且仅由小写字母构成，字符串不为空，并且字符串两两不同。整数和字符串之间用一个空格隔开。

接下来 m 行，其中第 i 行包含两个整数 a_i, s_i，表示第 i 条指令。若 $a_i = 0$，表示向左数 s_i 个人；若 $a_i = 1$，表示向右数 s_i 个人。保证 a_i 不会出现其他的数，$1 \le s_i < n$。

【输出格式】

输出到文件 ***toy.out*** 中。输出一个字符串，表示从第一个读入的小人开始，依次数完 m 条指令后到达的小人的职业。

【样例 1 输入】

```
7 3
0 singer
0 reader
0 mengbier
1 thinker
1 archer
0 writer
1 mogician
0 3
1 1
0 2
```

【样例 1 输出】

```
writer
```

【样例 1 说明】

这组数据就是【题目描述】中提到的例子。

【样例 2 输入】

```
10 10
1 c
0 r
0 p
1 d
1 e
1 m
1 t
1 y
1 u
0 v
1 7
1 1
1 4
0 5
0 3
0 1
1 6
1 2
0 8
0 4
```

【样例 2 输出】

```
y
```

【子任务】

子任务会给出部分测试数据的特点。如果你在解决题目中遇到了困难，可以尝试只解决一部分测试数据。

每个测试点的数据规模及特点如下表：

测试点	n	m	全朝内	全左数	$s_i = 1$	职业长度为 1
1	$= 20$	$= 10^3$	√	√	√	√
2			×			
3			√	×		
4			×			
5			√	√	×	
6			×			
7			√	×		
8			×			
9			√	√	√	×
10			×			
11			√	×		
12			×			
13			√	√	×	
14			×			
15			√	×		
16			×			
17	$= 10^5$	$= 10^5$	√	√		
18			×			
19			√	×		
20			×			

其中一些简写的列意义如下：

· 全朝内：若为“√”，表示该测试点保证所有的玩具小人都朝向圈内；

· 全左数：若为“√”，表示该测试点保证所有的指令都向左数，即对任意的 $1 \le i \le m$，$a_i = 0$；

· $s_i = 1$：若为“√”，表示该测试点保证所有的指令都只数 1 个，即对任意的 $1 \le i \le m$，$s_i = 1$；

· 职业长度为 1：若为“√”，表示该测试点保证所有玩具小人的职业一定是一个长度为 1 的字符串。

天天爱跑步（running）

【问题描述】

小 C 同学认为跑步非常有趣，于是决定制作一款叫做《天天爱跑步》的游戏。《天 天爱跑步》是一个养成类游戏，需要玩家每天按时上线，完成打卡任务。

这个游戏的地图可以看作一棵包含 n 个结点和 $n-1$ 条边的树，每条边连接两个结点，且任意两个结点存在一条路径互相可达。树上结点编号为从 1 到 n 的连续正整数。

现在有 m 个玩家，第 i 个玩家的起点为 S_i，终点为 T_i。每天打卡任务开始时，所有玩家**在第 0 秒**同时从**自己的起点**出发，以**每秒跑一条边**的速度，不间断地沿着最短 路径向着**自己的终点**跑去，跑到终点后该玩家就算完成了打卡任务。（由于地图是一棵树，所以每个人的路径是唯一的）

小 C 想知道游戏的活跃度，所以在每个结点上都放置了一个观察员。在结点 j 的观 察员会选择在第 W_j 秒观察玩家，一个玩家能被这个观察员观察到当且仅当该玩家在第 W_j 秒也**正好**到达了结点 j。小 C 想知道每个观察员会观察到多少人？

注意：我们认为一个玩家到达自己的终点后该玩家就会结束游戏，他不能等待一段时间后再被观察员观察到。即对于把结点 j 作为终点的玩家：若他在第 W_j 秒**前**到达终点，则在结点 j 的观察员**不能观察到**该玩家；若他**正好**在第 W_j 秒到达终点，则在结点 j 的观察员**可以观察到**这个玩家。

【输入格式】

从文件 ***running.in*** 中读入数据。

第一行有两个整数 n 和 m。其中 n 代表树的结点数量，同时也是观察员的数量，m 代表玩家的数量。

接下来 $n-1$ 行每行两个整数 u 和 v，表示结点 u 到结点 v 有一条边。 接下来一行 n 个整数，其中第 j 个整数为 W_j，表示结点 j 出现观察员的时间。 接下来 m 行，每行两个整数 S_i 和 T_i，表示一个玩家的起点和终点。 对于所有的数据，保证 $1 \le S_i, T_i \le n$， $0 \le W_j \le n$。

【输出格式】

输出到文件 ***running.out*** 中。

输出 1 行 n 个整数，第 j 个整数表示结点 j 的观察员可以观察到多少人。

【样例 1 输入】

```
6 3
2 3
```

1 2
1 4
4 5
4 6
0 2 5 1 2 3
1 5
1 3
2 6

【样例 1 输出】

```
2 0 0 1 1 1
```

【样例 1 说明】

对于 1 号点，$W_1 = 0$，故只有起点为 1 号点的玩家才会被观察到，所以玩家 1 和玩家 2 被观察到，共 2 人被观察到。

对于 2 号点，没有玩家在第 2 秒时在此结点，共 0 人被观察到。 对于 3 号点，没有玩家在第 5 秒时在此结点，共 0 人被观察到。 对于 4 号点，玩家 1 被观察到，共 1 人被观察到。

对于 5 号点，玩家 2 被观察到，共 1 人被观察到。

对于 6 号点，玩家 3 被观察到，共 1 人被观察到。

【样例 2 输入】

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

【样例 2 输出】

```
1 2 1 0 1
```

【子任务】

每个测试点的数据规模及特点如下表所示。提示：数据范围的个位上的数字可以帮助判断是哪一种数据类型。

测试点编号	n	m	约定
1	= 991	= 991	所有人的起点等于自己的终点，即 $S_i = T_i$
2			
3	= 992	= 992	$W_j = 0$
4			
5	= 993	= 993	无
6	= 99994	= 99994	树退化成一条链，其中 1 与 2 有边，2 与 3 有边，…，$n-1$ 与 n 有边
7			
8			
9	= 99995	= 99995	所有的 $S_i = 1$
10			
11			
12			
13	= 99996	= 99996	所有的 $T_i = 1$
14			
15			
16			
17	= 99997	= 99997	无
18			
19			
20	= 299998	= 299998	

【提示】

如果你的程序需要用到较大的栈空间（这通常意味着需要较深层数的递归），请务必仔细阅读选手目录下的文档 ***running/stack.pdf*** ，以了解在最终评测时栈空间的限制与在当前工作环境下调整栈空间限制的方法。

换教室（classroom）

【问题描述】

对于刚上大学的牛牛来说，他面临的第一个问题是如何根据实际情况申请合适的课程。

在可以选择的课程中，有 $2n$ 节课程安排在 n 个时间段上。 在第 i ($1 \le i \le n$ ）个 时间段上，两节内容相同的课程同时在不同的地点进行，其中，牛牛预先被安排在教室 ci 上课，而另一节课程在教室 di 进行。

在不提交任何申请的情况下，学生们需要按时间段的顺序依次完成所有的 n 节安排好的课程。如果学生想更换第 i 节课程的教室，则需要提出申请。若申请通过，学生就可以在第 i 个时间段去教室 di 上课，否则仍然在教室 ci 上课。由于更换教室的需求太多，申请不一定能获得通过。 通过计算，牛牛发现申请更 换第 i 节课程的教室时，申请被通过的概率是一个已知的实数 ki ，并且对于不同课程 的申请，被通过的概率是互相独立的。

学校规定，所有的申请只能在学期开始前一次性提交，并且每个人只能选择至多 m 节课程进行申请。 这意味着牛牛必须一次性决定是否申请更换每节课的教室，而不能根据某些课程的申请结果来决定其他课程是否申请；牛牛可以申请自己最希望更换教室的 m 门课程，也可以不用完这 m 个申请的机会，甚至可以一门课程都不申请。

因为不同的课程可能会被安排在不同的教室进行，所以牛牛需要利用课间时间从一间教室赶到另一间教室。

牛牛所在的大学有 v 个教室，有 e 条道路。每条道路连接两间教室，并且是可以双向通行的。

由于道路的长度和拥堵程度不同，通过不同的道路耗费的体力可能会有所不同。当第 i （ $1 \le i \le n-1$ ）节课结束后，牛牛就会从这节课的教室出发，选择一条耗费体力最少的路径前往下一节课的教室。现在牛牛想知道，申请哪几门课程可以使他因在教室间移动耗费的体力值的总和的期望值最小，请你帮他求出这个最小值。

现在牛牛想知道，申请哪几门课程可以使他因在教室间移动耗费的体力值的总和的期望值最小，请你帮他求出这个最小值。

【输入格式】

从文件 ***classroom.in*** 中读入数据。

第一行四个整数 n, m, v, e 。 n 表示这个学期内的时间段的数量； m 表示牛牛最多可以申请更换多少节课程的教室； v 表示牛牛学校里教室的数量； e 表示牛牛的学校里道路的数量。

第二行 n 个正整数，第 i （ $1 \le i \le n$ ）个正整数表示 c_i ，即第 i 个时间段牛牛被安排上课的教室；保证 $1 \le c_i \le v$ 。

第三行 n 个正整数，第 i （ $1 \le i \le n$ ）个正整数表示 d_i ，即第 i 个时间段另一间上同样课程的教室；保证 $1 \le d_i \le v$ 。

第四行 n 个实数，第 i（$1 \le i \le n$）个实数表示 k_i，即牛牛申请在第 i 个时间段更换教室获得通过的概率。保证 $0 \le k_i \le 1$。

接下来 e 行，每行三个正整数 a_j， b_j， w_j，表示有一条双向道路连接教室 a_j，b_j，通过这条道路需要耗费的体力值是 w_j；保证 $1 \le a_j, b_j \le v$， $1 \le w_j \le 100$。

保证 $1 \le n \le 2000$， $0 \le m \le 2000$， $1 \le v \le 300$， $0 \le e \le 90000$。保证通过学校里的道路，从任何一间教室出发，都能到达其他所有的教室。保证输入的实数最多包含 3 位小数。

【输出格式】

输出到文件 ***classroom.out*** 中。输出一行，包含一个实数，四舍五入精确到小数点后恰好 2 位，表示答案。你的输出必须和标准输出完全一样才算正确。

测试数据保证四舍五入后的答案和准确答案的差的绝对值不大于 4×10^{-3}。（如果你不知道什么是浮点误差，这段话可以理解为：对于大多数的算法，你可以正常地使用浮点数类型而不用对它进行特殊的处理）

【样例 1 输入】

```
3 2 3 3
2 1 2
1 2 1
0.8 0.2 0.5
1 2 5
1 3 3
2 3 1
```

【样例 1 输出】

```
2.80
```

【样例 1 说明】

所有可行的申请方案和期望收益如下表：

申请更换教室的时间段	申请通过的时间段	出现的概率	耗费的体力值	耗费的体力值的期望
无	无	1.0	8	8.0
1	1	0.8	4	4.8
	无	0.2	8	
2	2	0.2	0	6.4
	无	0.8	8	
3	3	0.5	4	6.0
	无	0.5	8	
1、2	1、2	0.16	4	4.48
	1	0.64	4	
	2	0.04	0	
	无	0.16	8	
1、3	1、3	0.4	0	2.8
	1	0.4	4	
	3	0.1	4	
	无	0.1	8	
2、3	2、3	0.1	4	5.2
	2	0.1	0	
	3	0.4	4	
	无	0.4	8	

【样例 2】

见选手目录下的 ***classroom/classroom2.in*** 与 ***classroom/classroom2.ans***。

【提示】

1. 道路中可能会有**多条**双向道路连接相同的两间教室。也有可能有道路两端连接的是**同一间**教室。
2. 请注意区分 n, m, v, e 的意义， n **不是**教室的数量， m **不是**道路的数量。

【子任务】

测试点	n	m	v	特殊性质 1	特殊性质 2
1	≤ 1	≤ 1	≤ 300	×	×
2	≤ 2	≤ 0	≤ 20		
3		≤ 1	≤ 100		
4		≤ 2	≤ 300		
5	≤ 3	≤ 0	≤ 20	√	√
6		≤ 1	≤ 100		×
7		≤ 2	≤ 300	×	
8	≤ 10	≤ 0		√	√
9		≤ 1	≤ 20		×
10		≤ 2	≤ 100	×	
11		≤ 10	≤ 300		√
12	≤ 20	≤ 0	≤ 20	√	×
13		≤ 1	≤ 100	×	
14		≤ 2	≤ 300	√	
15		≤ 20		×	√
16	≤ 300	≤ 0	≤ 20		×
17		≤ 1	≤ 100		
18		≤ 2	≤ 300	√	√
19		≤ 300		×	
20	≤ 2000	≤ 0	≤ 20		×
21		≤ 1			
22		≤ 2	≤ 100		
23		≤ 2000			
24			≤ 300		
25					

特殊性质 1：图上任意两点 a_i， b_i， $a_i \neq b_i$ 间，存在一条耗费体力最少的路径只包含一条道路。

特殊性质 2：对于所有的 $1 \leq i \leq n$， $k_i = 1$。

第 22 届全国青少年信息学奥林匹克联赛

CCF-NOIP-2016

提高组（复赛） 第二试

竞赛时间：2016 年 11 月 20 8:30 ~ 12:00

题目名称	组合数问题	蚯蚓	愤怒的小鸟
题目类型	传统型	传统型	传统型
目录	problem	earthworm	angrybirds
可执行文件名	problem	earthworm	angrybirds
输入文件名	problem.in	earthworm.in	angrybirds.in
输出文件名	problem.out	earthworm.out	angrybirds.out
每个测试点时限	1.0 秒	1.0 秒	2.0 秒
内存限制	512 MB	512 MB	512 MB
测试点数目	20	20	20
每个测试点分值	5	5	5

提交源程序文件名

对于 C++ 语言	problem.cpp	earthworm.cpp	angrybirds.cpp
对于 C 语言	problem.c	earthworm.c	angrybirds.c
对于 Pascal 语言	problem.pas	earthworm.pas	angrybirds.pas

编译选项

对于 C++ 语言	-lm	-lm	-lm
对于 C 语言	-lm	-lm	-lm
对于 Pascal 语言			

注意事项：

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. 除非特殊说明，结果比较方式均为忽略行末空格及文末回车的全文比较。
3. 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。
4. 全国统一评测时采用的机器配置为：CPU AMD Athlon(tm) II x2 240 processor，2.8GHz，内存 4G，上述时限以此配置为准。
5. 只提供 Linux 格式附加样例文件。
6. 评测在 NOI Linux 下进行。
7. 编译时不打开任何优化选项。

组合数问题（problem）

【问题描述】

组合数 C_n^m 表示的是从 n 个物品中选出 m 个物品的方案数。举个例子，从 $(1,2,3)$ 三个物品中选择两个物品可以有 $(1,2),(1,3),(2,3)$ 这三种选择方法。根据组合数的定义，我们可以给出计算组合数 C_n^m 的一般公式：

$$C_n^m = \frac{n!}{m!(n-m)!}$$

其中 $n! = 1 \times 2 \times \cdots \times n$。

小葱想知道如果给定 n, m 和 k，对于所有的 $0 \le i \le n, 0 \le j \le \min(i, m)$ 有多少对 (i, j) 满足 C_i^j 是 k 的倍数。

【输入格式】

从文件 ***problem.in*** 中读入数据。

第一行有两个整数 t, k，其中 t 代表该测试点总共有多少组测试数据，k 的意义见【问题描述】。

接下来 t 行每行两个整数 n, m，其中 n, m 的意义见【问题描述】。

【输出格式】

输出到文件 ***problem.out*** 中。

t 行，每行一个整数代表所有的 $0 \le i \le n, 0 \le j \le \min(i, m)$ 中有多少对 (i, j) 满足 C_i^j 是 k 的倍数。

【样例 1 输入】

```
1 2
3 3
```

【样例 1 输出】

```
1
```

【样例 1 说明】

在所有可能的情况中，只有 $C_2^1 = 2$ 是 2 的倍数。

【样例 2 输入】
```
2 5
4 5
6 7
```

【样例 2 输出】
```
7
```

【子任务】

测试点	n	m	k	t
1	≤ 3	≤ 3	= 2	= 1
2			= 3	≤ 10^4
3	≤ 7	≤ 7	= 4	= 1
4			= 5	≤ 10^4
5	≤ 10	≤ 10	= 6	= 1
6			= 7	≤ 10^4
7	≤ 20	≤ 100	= 8	= 1
8			= 9	≤ 10^4
9	≤ 25	≤ 2000	= 10	= 1
10			= 11	≤ 10^4
11	≤ 60	≤ 20	= 12	= 1
12			= 13	≤ 10^4
13	≤ 100	≤ 25	= 14	= 1
14			= 15	≤ 10^4
15		≤ 60	= 16	= 1
16			= 17	≤ 10^4
17	≤ 2000	≤ 100	= 18	= 1
18			= 19	≤ 10^4
19		≤ 2000	= 20	= 1
20			= 21	≤ 10^4

蚯蚓（earthworm）

【问题描述】

本题中，我们将用符号 $\mathrm{L}c\mathrm{J}$ 表示对 c 向下取整，例如：$\mathrm{L}3.0\mathrm{J}=\mathrm{L}3.1\mathrm{J}=\mathrm{L}3.9\mathrm{J}=3$。

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 只匠蚓的长度为 $a_i\ (i=1,2,\ldots,n)$，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 $\mathrm{L}p x\mathrm{J}$ 和 $x-\mathrm{L}p x\mathrm{J}$ 的匠蚓。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***

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：需要按从大到小的页序，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 $n, m \leq 1,000$。
- 测试点 8 ~ 14 满足 $q = 0$，其中测试点 8 ~ 9 还满足 $m \leq 10^5$。
- 测试点 15 ~ 18 满足 $m \leq 3 \times 10^5$。
- 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 $v \leq 2$，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。

<table>
<tr><th>测试点</th><th>n</th><th>m</th><th>t</th><th>a_i</th><th>v</th><th>q</th></tr>
<tr><td>1</td><td>$= 1$</td><td rowspan="3">$= 0$</td><td rowspan="8">$= 1$</td><td rowspan="12">$\le 10^6$</td><td rowspan="13">≤ 2</td><td rowspan="3">$= 0$</td></tr>
<tr><td>2</td><td>$= 10^3$</td></tr>
<tr><td>3</td><td>$= 10^5$</td></tr>
<tr><td>4</td><td>$= 1$</td><td rowspan="4">$= 10^3$</td></tr>
<tr><td>5</td><td>$= 10^3$</td></tr>
<tr><td>6</td><td>$= 1$</td><td rowspan="2">≤ 200</td></tr>
<tr><td>7</td><td>$= 10^3$</td></tr>
<tr><td>8</td><td>$= 5 \times 10^4$</td><td>$= 5 \times 10^4$</td><td rowspan="7">$= 0$</td></tr>
<tr><td>9</td><td rowspan="5">$= 10^5$</td><td>$= 10^5$</td><td>$= 2$</td></tr>
<tr><td>10</td><td>$= 2 \times 10^6$</td><td>$= 21$</td></tr>
<tr><td>11</td><td>$= 2.5 \times 10^6$</td><td>$= 26$</td></tr>
<tr><td>12</td><td>$= 3.5 \times 10^6$</td><td>$= 36$</td><td rowspan="2">$\le 10^7$</td></tr>
<tr><td>13</td><td>$= 5 \times 10^6$</td><td>$= 51$</td><td rowspan="2">$\le 10^9$</td></tr>
<tr><td>14</td><td>$= 7 \times 10^6$</td><td>$= 71$</td><td rowspan="8">$\le 10^8$</td></tr>
<tr><td>15</td><td rowspan="2">$= 5 \times 10^4$</td><td>$= 5 \times 10^4$</td><td>$= 1$</td><td rowspan="2">≤ 2</td><td rowspan="8">≤ 200</td></tr>
<tr><td>16</td><td>$= 1.5 \times 10^5$</td><td>$= 2$</td></tr>
<tr><td>17</td><td rowspan="4">$= 10^5$</td><td>$= 10^5$</td><td>$= 3$</td><td rowspan="4">$\le 10^9$</td></tr>
<tr><td>18</td><td>$= 3 \times 10^5$</td><td>$= 4$</td></tr>
<tr><td>19</td><td>$= 3.5 \times 10^6$</td><td>$= 36$</td></tr>
<tr><td>20</td><td>$= 7 \times 10^6$</td><td>$= 71$</td></tr>
</table>

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 = ax2 + 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 (xi, yi) 。

如果某只小鸟的飞行轨迹经过了 (xi, yi) ，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 (xi, yi) ，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 = −x2 + 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「$n/3+1$¬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 Ln/3J 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 「c¬ 和 LcJ 分别表示对 c 向上取整和向下取整，例如：「2.1¬ = 「2.9¬ =「3.0¬ = L3.0J = L3.1J = L3.9J = 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 = −x2 + 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 = −x2 + 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	