

第 31 届全国青少年信息学奥林匹克冬令营

竞赛时间：2014 年 2 月 13 日 8:00-13:00

题目名称	时空穿梭	紫荆花之恋	非确定机
目录	space	flower	nm
可执行文件名	space	flower	N/A
输入文件名	space.in	flower.in	nm1.in~ nm10.in
输出文件名	space.out	flower.out	nm1.out~ nm10.out
每个测试点时限	1 秒	12 秒	N/A
内存限制	512 MB	512 MB	N/A
测试点数目	10	20	10
每个测试点分值	10	5	10
是否有部分分	否	否	是
题目类型	传统型	传统型	提交答案型
是否有附加文件	否	否	是

提交源程序须加后缀

对于 C++ 语言	space.cpp	flower.cpp	N/A
对于 C 语言	space.c	flower.c	N/A
对于 Pascal 语言	space.pas	flower.pas	N/A

编译开关

对于 C++ 语言	-lm	-O2 -lm	N/A
对于 C 语言	-lm	-O2 -lm	N/A
对于 Pascal 语言		-O2	N/A

时空穿梭

【问题描述】

小 X 驾驶着他的飞船准备穿梭过一个 n 维空间，这个空间里每个点的坐标可以用 n 个实数来表示，即 $(x_1, x_2, \dots, x_n)$ 。

为了穿过这个空间，小 X 需要在这个空间选取 c ($c \geq 2$) 个点作为飞船停留的地方，而这些点需要满足以下三个条件：

1. 每个点的每一维坐标均为正整数，且第 i 维坐标不超过 m_i 。
2. 第 $i+1$ ($1 \leq i < c$) 个点的第 j ($1 \leq j \leq n$) 维坐标必须严格大于第 i 个点的第 j 维坐标。
3. 存在一条直线经过所选的所有点。在这个 n 维空间里，一条直线可以用 $2n$ 个实数 $p_1, p_2, \dots, p_n, v_1, v_2, \dots, v_n$ 表示。直线经过点 $(x_1, x_2, \dots, x_n)$ ，当且仅当存在实数 t ，使得对 $i = 1 \dots n$ 均满足 $x_i = p_i + tv_i$ 。

小 X 还没有确定他的最终方案，请你帮他计算一下一共有多少种不同的方案满足他的要求。由于答案可能会很大，你只需要输出答案 mod 10 007 后的值。

【输入格式】

输入文件 *space.in* 的第一行包含一个正整数 T ，表示有 T 组数据需要求解。

每组数据包含两行，第一行包含两个正整数 n, c ($c \geq 2$)，分别表示空间的维数和需要选择的暂停点个数。

第二行包含 n 个正整数，依次表示 $m_1, m_2, \dots, m_n$ 。

【输出格式】

输出文件 *space.out* 包含 T 行，每行包含一个非负整数，依次对应每组数据的答案。

【样例输入 1】

```
3
2 3
3 4
3 3
3 4 4
4 4
5 9 7 8
```

【样例输出 1】

2
4
846

【样例说明】

样例数据第一组共有两种可行方案：一种是选择 (1,1), (2,2), (3,3)，另一种是选择 (1,2), (2,3), (3,4)。

【样例输入输出 2】

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

【数据规模】

测试点编号	T	n	c	m_i
1	= 1,000	= 1	≤ 20	$\leq 100,000$
2	= 3	≤ 4	≤ 20	≤ 30
3	= 3	= 2	= 3	$\leq 100,000$
4	= 1,000	= 2	= 3	$\leq 100,000$
5	= 20	≤ 5	= 3	$\leq 100,000$
6	= 100	≤ 11	= 3	$\leq 100,000$
7	= 1	≤ 5	≤ 20	$\leq 100,000$
8	= 20	≤ 5	≤ 20	$\leq 100,000$
9	= 100	≤ 11	≤ 20	$\leq 100,000$
10	= 100	≤ 11	≤ 20	$\leq 100,000$

紫荆花之恋

【问题描述】

强强和萌萌是一对好朋友。有一天他们在外闲逛，突然看到前方有一棵紫荆树。这已经是紫荆花飞舞的季节了，无数的花瓣以肉眼可见的速度从紫荆树上长了出来。

仔细看看的话，这个大树实际上是一个带权树。每个时刻它会长出一个新的叶子节点。每个节点上有一个可爱的小精灵，新长出的节点上也会同时出现一个新的小精灵。小精灵是很萌但是也很脆弱的生物，每个小精灵 i 都有一个感受能力值 r_i ，小精灵 i, j 成为朋友当且仅当在树上 i 和 j 的距离 $\text{dist}(i, j) \leq r_i + r_j$ ，其中 $\text{dist}(i, j)$ 表示在这个树上从 i 到 j 的唯一路径上所有边的边权和。

强强和萌萌很好奇每次新长出一个叶子节点之后，这个树上总共有几对朋友。

我们假定这个树一开始为空，节点按照加入的顺序从 1 开始编号。由于强强非常好奇，你必须在他每次出现新节点后马上给出总共的朋友对数，不能拖延哦。

【输入格式】

输入文件 *flower.in* 共有 $n + 2$ 行。

第一行包含一个正整数，表示测试点编号。

第二行包含一个正整数 n ，表示总共要加入的节点数。

我们令加入节点前的总共朋友对数是 last_ans ，在一开始时它的值为 0。

接下来 n 行中第 i 行有三个数 a_i, c_i, r_i ，表示节点 i 的父节点的编号为 $a_i \text{ xor } (\text{last_ans} \bmod 10^9)$ (其中 xor 表示异或， $\bmod$ 表示取余，数据保证这样操作后得到的结果介于 1 到 $i - 1$ 之间)，与父节点之间的边权为 c_i ，节点 i 上小精灵的感受能力值为 r_i 。

注意 $a_1 = c_1 = 0$ ，表示 1 号点是根节点，对于 $i > 1$ ，父节点的编号至少为 1。

【输出格式】

输出文件 *flower.out* 包含 n 行，每行输出 1 个整数，表示加入第 i 个点之后，树上有多少对朋友。

【样例输入】

```
0
5
0 0 6
```

```
1 2 4
0 9 4
0 5 5
0 2 4
```

【样例输出】

```
0
1
2
4
7
```

【样例输入输出 2】

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

【数据规模和约定】

对于所有的数据，满足 $1 \leq c_i \leq 10000, a_i \leq 2 * 10^9, r_i \leq 10^9$

测试点编号	约定
1,2	$n \leq 100$
3,4	$n \leq 1000$
5,6,7,8	$n \leq 100000$ ，节点 1 最多有两个子节点，其它节点最多有一个子节点
9,10	$n \leq 100000$ ， $r_i \leq 10$
11,12	$n \leq 100000$ ，这棵树是随机生成的
13,14,15	$n \leq 70000$
16,17,18,19,20	$n \leq 100000$

非确定机

【问题描述】

“非确定机”是现在假想中的一种计算机，它可以同时运行任意多段指令。这种计算机中允许一种新的分支指令，执行到这条指令时，程序会一分为二，同时分别执行这两个分支。

“非确定机”的一个神奇的功能是程序反转。给定一个程序和该程序的输出，它可以用相同的时空代价得到一个符合该输出的输入。这道题目正是要你反转运行一个程序。

在本题中，你有一个已编译好的，包含一些算法的程序 *prog*。它的输入为一个有向图 G 和算法编号 k ，输出为在有向图 G 上运行算法 k 得到的结果。同时，你会得到 10 个由程序 *prog* 运行得到的输出文件。你的任务是对每个输出文件给出一个可能的输入文件。

为了区分，在后文中如果没有特殊说明，我们把给定的程序 *prog* 的输出文件称作“输入”，把你需要提交的程序 *prog* 的输入文件称作“输出”。

【输入格式】

该题为提交答案型试题，所有输入数据 *nm1.in~nm10.in* 已在试题目录下。

输入的第一行包含三个正整数 n, k, p ，表示图 G 的点数，当前使用算法和一个由图 G 计算出的评分参数。

接下来有若干行，每行包含若干个整数，其意义需要你去探究。

【输出格式】

针对给定的 10 个输入文件 *nm1.in~nm10.in*，你需要分别提交你的输出文件 *nm1.out~nm10.out*。

输出文件的第一行包含 3 个整数 n, m, k ，表示有向图 G 的点数，边数和当前使用的算法。

接下来有 m 行，每行包含 3 个整数 u, v, w ，表示一条从节点 u 连向节点 v ，权值为 w 的有向边。其中节点的编号用 1 到 n 的整数表示。要求 w 必须为不超过 20000 的非负整数，且不能出现重边。允许出现自环。

【程序的使用】

在终端中先切换到该试题的目录下

```
cd nm
```

程序 *prog* 已在试题目录下，其使用方法是

```
./prog <output_file>
```

程序会把 *<output_file>* 作为程序 *prog* 的输入，将运行的结果输出到标

准输出。例如

`./prog nm4.out`

程序将会把 `nm4.out` 作为程序 *prog* 的输入。

如果你的文件不合法，程序将会输出错误信息。

【样例输入】

```
3 0 0
0 0 1
1 0 0
0 1 0
```

【样例输出】

```
3 3 0
1 3 92
2 1 12
3 2 29
```

【样例说明】

在样例中，边的权值 w 并不会影响到运行结果，在实际的数据中也要注意可能存在这种情况。

【评分方法】

每个测试点单独评分。

如果你的输出运行后得到的 n, k 与输入文件一致，得 1 分。

如果你的输出运行后得到的 n, k, p 与输入文件一致，得 2 分。

如果你的输出运行后得到的整数中，除了 p 以外均与输入文件一致，得 4 分。

在评测时，每个测试点会有一个评分参数 s 。如果你的输出运行后得到的整数中，除了 p 以外均与输入文件一致，且 p 值与标准文件差值的绝对值不超过 s ，得 7 分。

如果你的输出运行后得到的整数和输入文件全部一致，得 10 分。

以上条件如果满足多个，取最高分。

每个测试点的 s 如下

测试点编号	s	测试点编号	s
1	0	6	8
2	1000	7	20
3	0	8	1000

4	1000	9	1000
5	0	10	0

【如何测试你的输出】

程序 `prog` 还有测试的功能，可以根据你的输入输出文件以及 s 值给出得分或错误信息。具体用法为

```
./prog <output_file> <input_file> <s>
```

例如要测试测试点 6，可以使用

```
./prog nm6.out nm6.in 8
```

【提示】

题目中大多数 k 的值由两位数构成，若两种算法 k 的第一位相同，表示这两种算法采用了相同的基本算法。

请妥善保存输入文件 `nm*.in` 和你的输出 `nm*.out`，及时备份，以免误删。