凉心模拟 Day2

DL24

题目名称	锻造	整除	欠钱
源程序文件名	forging.pas/c/cpp	division.pas/c/cpp	money.pas/c/cpp
输入文件名	forging.in	division.in	money.in
输出文件名	forging.out	division.out	money.out
每个测试点时限	1s	2.5s	1s
内存限制	256MB	256MB	512MB
测试点数目	10	10	10
每个测试点分值	10	10	10
题目类型	传统型	传统型	传统型
是否有附加文件	否	是	否

评测机配置:
Ubuntu 16.04 LTS
Intel Core i5-4590 CPU @ 3.7GHz
RAM 3864MiB
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1 16.04.4)
自行本地评测时，时间限制以标程最慢点两倍为准。
对于输入数据较多的题目，请使用较为快速的读入方式。

1 锻造 (forging)

1.1 题目背景

勇者虽然武力值很高，但在经历了多次战斗后，发现怪物越来越难打，于是开始思考是不是自己平时锻炼没到位，于是苦练一个月后发现……自己连一个史莱姆都打不过了。

勇者的精灵路由器告诉勇者其实是他自己的武器不好，并把他指引到了锻造厂。

1.2 题目描述

“欢迎啊，老朋友。”

一阵寒暄过后，厂长带他们参观了厂子四周，并给他们讲锻造的流程。

“我们这里的武器分成若干的等级，等级越高武器就越厉害，并且对每一等级的武器都有两种属性值 b 和 c，但是我们初始只能花 a 个金币来生产 1 把 0 级剑……”

“所以你们厂子怎么这么垃圾啊，不能一下子就造出来 999 级的武器吗?”勇者不耐烦的打断了厂长的话。

“别着急，还没开始讲锻造呢……那我们举例你手中有一把 x 级武器和一把 y 级武器 ($y = max(x-1,0)$)，我们令锻造附加值 $k = min(c_x, b_y)$，则你有 $\frac{k}{c_x}$ 的概率将两把武器融合成一把 $x+1$ 级的武器。”

“……但是，锻造不是一帆风顺的，你同样有 $1-\frac{k}{c_x}$ 的概率将两把武器融合成一把 $max(x-1,0)$ 级的武器……”

勇者听完后暗暗思忖，他知道厂长一定又想借此机会坑骗他的零花钱，于是求助这个村最聪明的智者——你，来告诉他，想要强化出一把 n 级的武器，其期望花费为多少?

由于勇者不精通高精度小数，所以你只需要将答案对 998244353($7 \times 17 \times 2^{23}+1$，一个质数) 取模即可。

1.3 格式

1.3.1 输入格式

第一行两个整数 n, a，含义如题所示。

为了避免输入量过大，第二行五个整数 bx, by, cx, cy, p，按照下列代码来生成 b 和 c 数组。

```
b[0]=by+1;c[0]=cy+1;
for(int i=1;i<n;i++){
    b[i]=((long long)b[i-1]*bx+by)%p+1;
    c[i]=((long long)c[i-1]*cx+cy)%p+1;
}
```

1.3.2 输出格式

输出一行一个整数，表示期望花费。

1.4 样例

1.4.1 样例 1 输入

```
0 6432
4602677 3944535 2618884 6368297 9477531
```

1.4.2 样例 1 输出

```
6432
```

1.4.3 样例 2 输入

```
1 3639650
6136976 5520115 2835750 9072363 9302097
```

1.4.4 样例 2 输出

```
150643649
```

1.4.5 样例 3 输入

```
10 2
2 33 6 66 2333333
```

1.4.6 样例 3 输出

```
976750710
```

1.4.7 样例 4 输入

```
200 5708788
0 0 0 0 1
```

1.4.8 样例 4 输出

```
696441597
```

1.5 数据范围

测试点	n≤	特殊性质
1	0	N/A
2	1	
3	200	有
4		N/A
5	2000	有
6		N/A
7	1000000	有
8		N/A
9	10000000	有
10		N/A

对于特殊性质处标示为“有”的数据满足 $p = 1$。

对于 100% 的数据，$0 \leq a \leq 10^7, 0 \leq bx, by, cx, cy < p < 10^7, 0 \leq n \leq 10^7$

2 整除 (division)

2.1 题目描述

整除符号为 |，$d|n$ 在计算机语言中可被描述为 $n\%d == 0$。
现有一算式 $n|x^m - x$，给定 n，m，求 $[1, n]$ 以内 x 解的个数。
解可能很大，输出取模 998244353。

2.2 格式

2.2.1 输入格式

其中 n 的给定方式是由 c 个不超过 t 的质数的乘积给出的，c 和 t 的范围会在数据范围中给出。
第一行一个 id 表示这个数据点的标号。
多组数据，其中第二行一个整数 T 表示数据组数。
对于每一组数据:
第一行两个整数 c 和 m。
第二行 c 个整数，这些整数都是质数，且两两不同，他们的乘积即为 n。
由于你可以通过输入求出 t，输入不再给出。

2.2.2 输出格式

对于每组数据输出一行，表示解的个数。

2.3 样例

2.3.1 样例输入

```
0
1
2 3
2 3
```

2.3.2 样例输出

```
6
```

另有两个样例，见下发文件。

2.4 数据范围

测试点	$c \leq$	$t \leq$	$m \leq$	$T \leq$
1	2	10^3	2	50
2	2	10^3	10^9	50
3	2	10^2	10	10000
4	1	10^4	2	50
5	2	10^4	2	50
$6, 7, 8$	10	10^4	10^9	50
$9, 10$	50	10^4	10^9	50

其中所有数据点都满足 $1 \leq c \leq 50$,$1 \leq t \leq 10^4$,$1 \leq m \leq 10^9$,$1 \leq T \leq 10000$。

3 欠钱 (money)

3.1 题目描述

南极的企鹅王国大学中生活着 n 只企鹅，作为 21 世纪的优秀大学生，企鹅们积极响应“大众创业，万众创新”的号召，纷纷创业。但是创业需要资金，企鹅们最近手头比较紧，只能互相借钱。

企鹅的借钱行为是有规律可循的：每只企鹅只会借一次钱，并且只会从一只企鹅那里借钱。借钱关系中不存在环（即不存在类似“金企鹅欠银企鹅钱，银企鹅欠铜企鹅钱，铜企鹅欠金企鹅钱”这种情况）。

企鹅的还钱行为也是有规律可循的：每只企鹅一旦新获得了一笔钱，就会立刻用这笔钱尽可能偿还自己欠的债务，直到债务偿清或用光这笔钱。它只会使用新获得的这笔钱，至于以前它有没有钱、有多少钱，与还钱行为无关。

企鹅们经常会做美梦。在一只企鹅 A 的梦里，它梦见自己创业成功，一下子获得了 $+\infty$ 元钱，于是（按照上文的还钱规则）它赶快把钱用来还债，接着拿到钱的那只企鹅也赶快把钱用来还债……如此往复，直到所有获得钱的企鹅都完成了还债操作。梦醒之后，它开心地把梦的内容告诉了另外一只企鹅 B，企鹅 B 听了，也很开心，于是它问道：在你的梦里，我获得了多少钱呢？(指 B 去还债之前手里的钱，包括后来用于还债的钱和还债后 B 手里剩下的钱。)

梦毕竟是梦，对实际的欠债情况没有影响。

3.2 格式

3.2.1 输入格式

第一行两个整数 n 和 m，表示有 n 只企鹅，m 个操作。

接下来 m 行，有两种可能的格式：

- 0 a b c：修改操作，企鹅 a 向企鹅 b 借了 c 元钱。- 1 a b：查询操作，询问假如 a 有了 $+\infty$ 元钱，企鹅 b 会净收入多少钱。

本题强制在线，也就是说：对于每个操作输入的变量 a, b, c（如果没有 c，那就只有 a, b）都不是实际的 a, b, c，想获得实际的 a, b, c 应当经过以下操作：

```
a = (a + lastans) % n + 1;
b = (b + lastans) % n + 1;
c = (c + lastans) % n + 1;
```

其中，lastans 是上一次询问的答案。如果没有上一次询问，lastans 为 0。

3.2.2 输出格式

对每个询问操作，输出一行一个数表示答案。

3.3 样例

3.3.1 样例输入

```
5 9
0 1 2 1
0 0 1 2
1 0 1
1 2 4
0 2 1 1
1 2 0
0 3 1 0
1 4 2
1 3 4
```

3.3.2 样例输出

```
3
2
0
1
0
```

3.4 数据范围

数据分为以下几种：

第一种：占 10%，$n \leq 5000$ 且 $m \leq 10000$；

第二种：占 20%，所有借钱事件（0 开头的操作）发生在所有询问事件（1 开头的操作）之前；

第三种：占 30%，对于一只企鹅 A，最多只有一只企鹅向 A 借钱；

第四种：占 40%，没有特殊性质，n、m 大小有一定梯度。

对于所有数据，满足：$n \leq 10^5$，$m \leq 10^6$，$0 \leq a, b, c \leq n$ 且 $a \neq b$。