又一套随便 ak 的差题

STSO-Easy Round-2

——A Stupid Test by a Stupid Oier l1ll5

题目名称	分裂	差题	偷窃
英文名称	split	bad	steal
目录	split	bad	steal
可执行文件名	split.cpp	bad	steal.cpp
输入文件名	split.in	bad.in	steal.in
输出文件名	split.out	bad.out	steal.out
每个测试点时限	1 秒	1 秒	1 秒
内存限制	256 MB	256 MB	256 MB
试题总分	100	100	100
测试点数目	10	10	10
每个测试点分值	10	10	10
是否有部分分	否	否	否
题目类型	传统型	传统型	传统型

提交源程序须加后缀

对于 C++ 语言	split.cpp	bad.cpp	steal.cpp
对于 C 语言	不资词	不资词	不资词
对于 Pascal 语言	不资词	不资词	不资词

注意：最终测试时，所有编译命令均不打开任何优化开关。

另外，在考场上 AK 掉 STSO-Easy Round 系列中一套题的选手（若有多人，取总用时最短者）可以获得出题人提供的小奖 δ 品 :一套正版 wallpaper engine（steam 国区礼物）。

出题人联系方式：QQ 2817629709 （支持在线答疑

个人博客：http://www.zgz233.xyz/（将会存放各系列题目的 pdf 和 sol

分裂

【问题描述】

爆裂吧，现实！粉碎吧，精神！Banishiment this world！——

我们都知道，在这个世界上有着一种疯狂的兔子！它们从两岁开始就会不断的分裂，每年分裂出一只新的0岁的小兔子，它们永远都不会死亡，只会以极快的速度增长。

如果我们在第1年有一只1岁的兔子，想要知道到了第n年会有多少只兔子。这将会是一个简单的问题。

然而，这世上除了【数据删除】以外没谁能够永生，所以这些兔子将会在10岁的那次分裂后死掉。现在，希望你能够回答这个依旧很简单的问题。

为了方便，你只需要输出答案对1000000007取模后的结果即可。

【输入格式】

从文件 ***split.in*** 中输入数据。

输入的第一行包含一个正整数T，表示有T组询问。

接下来的T行每行包含一个正整数n，表示你需要求出到了第n年会有多少只兔子。

【输出格式】

输出到文件 ***split.out*** 中。

输出T行，每行一个正整数，表示第n年兔子数对1000000007取模的值。

【样例输入】

```
5
1
3
10
88
233
```

【样例输出】

```
1
3
88
352138150
```

```
199222855
```

【样例说明】

在此仅强调第10年的情况，若不考虑死亡，则答案应为89，但在第10年，第1年的一只兔子死亡。答案为88。

【数据规模与约定】

对于 100%的数据，满足 $1 \leq T \leq 10$，$1 \leq n \leq 10^{18}$

○：对于测试点 1，满足 $n = 11$

①：对于测试点 2,3,4，满足$n \leq 10^5$

②：对于测试点 5，满足$n = 6666666$

③：对于测试点 6,7，满足$n \leq 10^9$

④：对于测试点 8，满足$n = 233333333$

⑤：对于测试点 9,10，没有任何特殊限制。

差题

【问题描述】

这道题换了好多次，不写题面了。

在这道题中，你需要维护一个长度为n的整数序列 A，其中记第i个元素为a_i。

将会给出m次操作，对于每次操作，首先读入一个正整数opt，表示操作种类：

$opt=1$：读入一个正整数i，一个整数v，表示置$a_i=v$

$opt=2$：读入一个正整数i，一个整数v，表示置$a_i=a_i+v$

$opt=3$：读入一个正整数l，一个正整数r，表示置$a_i=-a_i$，其中，$l\le i\le r$

$opt=4$：读入一个正整数l，一个正整数r，输出a_l~a_r的最大值

$opt=5$：读入一个正整数l，一个正整数r，输出a_l~a_r的最小值

$opt=6$：读入一个正整数l，一个正整数r，输出a_l~a_r的和（其中 4、5、6 均包含l，r）

我们用某种方法保证了你程序的在线性：

对于每个$opt=1$、2，令你需要实际处理的序列下标为I，读入的下标为i，

则$I=(i\ xor\ lastans)mod\ n+1$。

对于每个$opt=3$、4、5、6，令你实际需要处理的区间为$[L,R]$，读入的区间为$[l,r]$，则$L=(l\ xor\ lastans)mod\ n+1$，$R=(r\ xor\ lastans)mod\ n+1$。此后，若不满足$L\le R$，交换它们。

其中，$lastans$表示上一次询问操作输出答案的**绝对值**，一开始，$lastans=0$

上述+1均在取模操作后执行。

【输入格式】

从文件 ***bad.in*** 中输入数据。

输入的第一行包含两个正整数n,m，分别表示序列长度和操作次数。

输入的第二行包含n个正整数，表示a_1~a_n的初始值。

接下来的m行，每行按照【问题描述】的格式给出。

【输出格式】

输出到文件 ***bad.out*** 中。

对于每个询问操作，输出这次询问的答案。

【样例输入】

```
5 6
1 2 3 4 5
1 3 1
5 2 3
2 1 1
4 1 5
3 2 3
6 1 4
```

【样例输出】

```
1
5
1
```

【样例说明】

在未执行任何操作时，原序列为 1,2,3,4,5
执行第一次操作后，序列为 1,2,3,1,5，这次操作实际修改下标 i=(3 xor 0)mod 5 +1=4
执行第二次操作后，序列为 1,2,3,1,5，输出 2,1,4 中最小值 1，此时修改 lastans=1
执行第三次操作后，序列为 2,2,3,1,5，这次操作实际修改下标 i=(1 xor 1) mod 5 +1 = 0+1=1
执行第四次操作后，序列为 2,2,3,1,5，这次操作实际询问区间[2,5]，最大值为 5，此时修改 lastans=5
执行第五次操作后，序列为 2,-2,-3,1,5，实际修改区间[2,3]
执行第六次操作后，序列为 2,-2,-3,1,5，实际询问区间[2,4]，此时修改 lastans=1

【数据规模与约定】

对于 100%的数据，满足 $1 \leq n, m \leq 200000$，初始 $-10^3 \leq a_i \leq 10^3$， $-10^3 \leq v \leq 10^3$

○：对于测试点 1，满足只存在$opt = 6$

①：对于测试点 2,3，满足 $1 \leq n, m \leq 5000$

②：对于测试点 4,5,6，不存在$opt = 3$

③：对于测试点 7，满足只存在操作 3,6，且所有操作 3 在操作 6 前。

④：对于测试点 8,9,10，不具有任何特殊性质

下发的大样例满足测试点 2 的数据限制。

数据由构造和随机混合生成。

良心提示：think twice , code once .

偷窃

【问题描述】

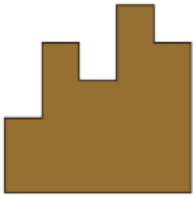
我们都知道，仅仅通过三视图去观察一个物体并不能把它的实际情况唯一确定。不过显然，脑小的 everlasting 并不知道这一点，这当然就给了传奇怪盗 starria 可乘之机！

everlasting 被要求去看守 SBC 银行（Stupid Bank of China）的地下金库，在地下金库，黄金都被切割成了一些 1*1*1 大小的金砖，并在一个底面为 r*c，高度不限的长方体中存放。金砖都是被整齐地摆放的，同时它们受到重力的影响，也就是说不会有金砖悬在空中。（你可以结合后图理解金砖摆放的方式）

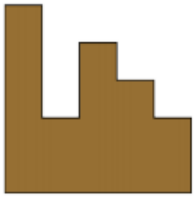
为了偷懒，everlasting 购买了三台摄像机，分别从：前方，左面，顶部三个方向对金砖进行监控（你可以理解为取得了金砖摆放的三视图），这样 everlasting 就只需要每隔一段时间看看三台摄像机的画面有没有变化就可以判断 starria 是否偷走了金砖。然而我们说过，这么做是存在漏洞的，starria 想问问你她最多可以在不被发现的情况下偷走几块金砖？

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

实际的金砖摆放示意图



从前方观察景象



从侧（左）面观察景象

从顶部观察景象

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

而当 starria 成功洗劫金库后，可能存在这么一种剩余金砖摆放使得 everlasting 无法发现金库惨遭洗劫。

【输入格式】

从文件 ***steal.in*** 中输入数据。

输入的第一行包含两个正整数r和c，表示金库网格的行列数。

接下来的r行，每行c个整数，其中第i行第j个数$h_{i,j}$表示第i行第j列上金砖的初始高度。

【输出格式】

输出到文件 ***steal.out*** 中。

输出一行一个正整数，表示 starria 在不被发现的情况下可能偷走的最多金砖数。

【样例输入】

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

【样例输出】

```
9
```

【样例说明】

原摆放方案需要 45 块金砖，【问题描述】中的附图揭示了一种只需要 36 块金砖的摆放方案。可以偷走 9 块金砖。

【数据规模与约定】

对于 100%的数据，满足$r \leq 100$，$c \leq 100$，$0 \leq h_{i,j} \leq 10^9$

○：对于测试点 1，满足$h_{i,j} = 0$

①：对于测试点 2,3，满足$r * c \leq 4$，$\sum h \leq 10$

②：对于测试点 4,5,6，满足$h_{i,j} \geq 1$

③：对于测试点 7,8,9,10，没有任何特殊限制

良心提示：测试点4,5,6的特殊限制具有可扩展性。

下发的大样例满足测试点 4 的数据限制。