NOIP 模拟赛

Cmd2001 2019.01.31

(请参赛选手务必仔细阅读本页内容)

1. 题目概况：

|  |  |  |  |
| --- | --- | --- | --- |
| 题目名称 | 块 | 拓展Xor | 弦与切 |
| 文件名称 | block | extxor | sinandcos |
| 输入文件名 | block.in | extxor.in | sinandcos.in |
| 输出文件名 | block.out | extxor.out | sinandcos.out |
| 时间限制 | 1s | 2s | 2s |
| 空间限制 | 256mb | 256mb | 256mb |
| 测试点数目 | 20 | 20 | 20 |
| 单个测试点分值 | 5 | 5 | 5 |
| 题目类型 | 传统型 | 传统型 | 传统型 |
| 是否有SPJ | 否 | 否 | 否 |
| 编译命令 | g++ block.cpp -o block | g++ extxor.cpp -o extxor | g++ sinandcos.cpp -o sinandcos |

1. 测试环境：

硬件环境：

软件环境：

本次测试不开启优化，不开启。

 块

(block)

**【问题描述】**

*「いては散り、散っては咲く桜のように 巡り巡る绿に誘われ*

*(开了又谢、谢了又开的樱花一样，循环的因缘之约)*

*来世できっとまた、お逢いしましょう*

*(来生，我们一定会再相逢)」*

*——《拝啓、御前様》*

传说在世界的尽头有一片巨大的樱花林，这里的每一棵樱树都会开出不同颜色的樱花。

而每一棵樱树开出的樱花，一定能构成一个联通的区块。

这里我们定义：两个块相邻当且仅当这两个块有且仅有一个维度的坐标的差值为。

现在，我们拿到了这片樱花林的地图，上面标注了每个格子樱花的花色。

我们需要一个程序去计算，在这片樱花林里，至少有多少棵樱树。

当然，由于世界内核的崩坏，这个世界的地图不一定是二维的。所以，你的程序需要处理更加复杂的情况。

**【输入描述】**

第一行一个整数，表示地图的维数。

接下来一行个整数，表示每一维的大小。

接下来个整数，按照从第维度到第维度依次递增循环的顺序给出地图。

即：地图按照以下伪代码给出：

**【输出描述】**

一行一个整数，表示所求答案。

**【样例输入】**

[样例1]

1

20

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

[样例2]

2

3 3

0 1 1

0 1 2

2 1 2

[样例3]

见下发文件

**【样例输出】**

[样例1]

17

[样例2]

4

[样例3]

见下发文件

**【数据范围及提示】**

对于的数据，；

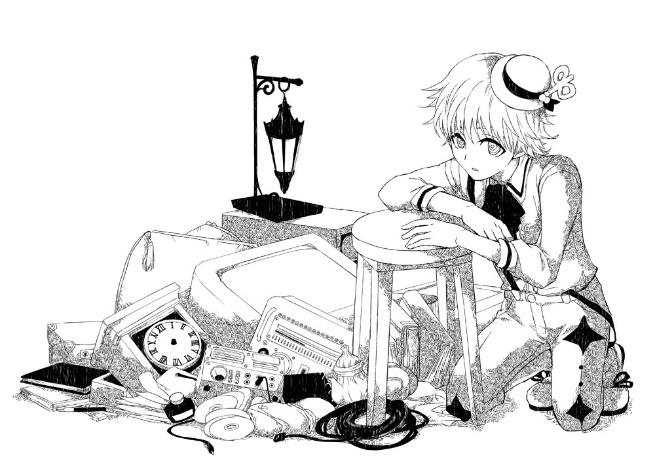
对于另外的数据，；

对于的数据，。

拓展Xor

(extxor)

**【问题描述】**

*「世界を忘れたくないならば*

*(你不愿忘却这个世界的话)*

*君が選んだ哀しい物語を消し*

*(我会将你所选择的悲伤故事消去)*

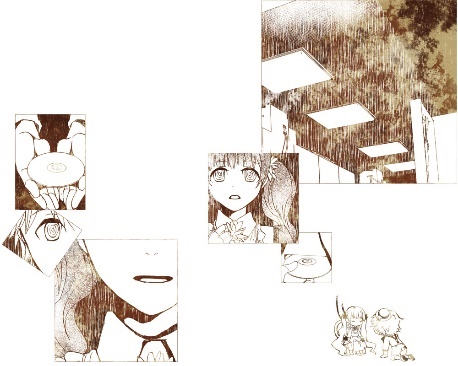
*記憶を書き換えたのは*

*(改写的记忆便是)*

*独りぼっちの君への贈り物*

*(我赠与孤身一人的你的礼物)」*

*——《Planetia》*

要想改写记忆的话，我们必须知道Lanjerd的系统密码。

对此，我们只有这样的线索：

有个巨大的整数和一个进制。

无限使用这些整数，进行进制下的不进位加法。

通过这种组合能拼凑出的最大数字，就是Lanjerd的密码了。

**【输入描述】**

第一行两个整数，表示数字个数和进制。

接下来行，每行一个十进制整数，表示一个给出的数字。

数字不保证没有前导零。

**【输出描述】**

一行一个十进制整数，表示所求的答案。

你不可以输出前导零。

**【样例输入】**

[样例1]

5 2

76

121

8

978

910

[样例2]

5 3

803

87

760

771

9

[样例3]

见下发文件

**【样例输出】**

[样例1]

1023

[样例2]

1698

[样例3]

见下发文件

**【数据范围及提示】**

对于的数据，；

对于的数据，；

对于的数据，；

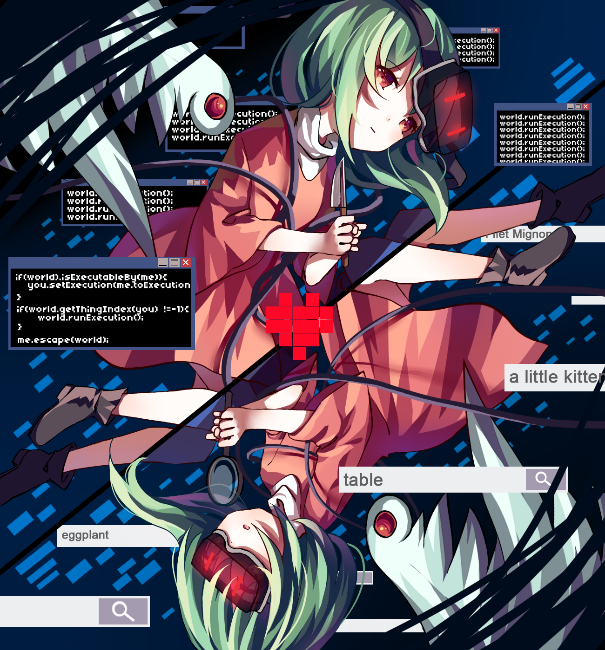
对于的数据， 。

保证为质数。

弦与切

(sinandcos)

**【问题描述】**

*「If I'm a sine wave*

*(如果我是一条正弦波)*

*Then you can sit on all my*

*(那么请你坐上)*

*TANGENTS*

*(我的切线)」*

*——《world.execute (me) ;》*

而我们今天的问题，就和这正弦函数有关。

现在我们有一个长度为的行向量(数组？)。

我会给出一个区间，你需要求出其正弦值的和。(即：)。

众所周知，求导是(即：()。

所以我还会给出一个区间，让你求出其正弦函数的切线的斜率(其实就是余弦函数)的和(即：。

当然，只有这两种操作还是过于无聊，所以有时我还会给出一个区间，让其中的每个数都加上一个浮点数。

**【输入描述】**

第一行两个整数，表示行向量长度和操作次数。

接下来一行个浮点数，表示行向量的初始值。

接下来行每行三个整数，表示第个操作的类型和区间的左右边界。

如果，表示你需要求出区间的正弦函数的和。

如果，表示你需要要求出区间的正弦函数的切线的斜率(其实就是余弦函数)的和。

如果，那么该行还会跟一个浮点数，表示你需要对区间中的每一个数都加上。

(注意这里的区间都是闭区间 ~~的说~~)

**【输出描述】**

对于每个询问操作，输出一行一个浮点数，表示询问的答案。

输出请采用浮点形式输出，保留三位小数，整数部分为或小数位数不足的用补齐，你的答案必须和标准答案完全相同才能得分。

（对于选手，可以用 进行输出）

**【样例输入】**

[样例1]

3 3

0 -2 -6

3 1 2 -7

2 1 2

1 2 3

[样例2]

7 3

0.123 -19.002 -57.507 34.434 -80.886 -17.115 69.843

2 2 7

3 2 7 16.341

1 1 3

[样例3]

见下发文件

**【样例输出】**

[样例1]

-1.069

0.049

[样例2]

1.519

-0.842

[样例3]

见下发文件

**【数据范围及提示】**

对于的数据，；

对于另外的数据，不包含修改操作；

对于的数据，。

输入中所有的浮点数大小均小于，且以三位小数形式给出。