C3 - Statement

A 奇怪的位运算 2023

题目描述

有这样一种奇怪的位运算 $\odot$，它将两个自然数二进制表示的每一位按照如下法则进行运算：

a_i	b_i	$(a \odot b)_i$
0	0	0
0	1	0
1	0	1
1	1	0

其中 $a_i, b_i, (a \odot b)_i$ 分别表示 $a, b, a \odot b$ 的二进制表示中的第 i 位。

你能迅速的给出两个数进行这种奇怪的位运算后的得数吗？

没有思路的话可以去看一下Hint~

输入

多组数据输入，每组数据为一行两个正整数 a, b，$0 \leq a, b \leq 10^9$，保证数据组数不超过 10^5

输出

对每组数据，输出一行一个数字，为 $a \odot b$ 的值

输入样例

```
7 10
14 19
```

输出样例

```
5
12
```

样例解释

- 样例1：7 的8位二进制为 00000111，10 的8位二进制为 00001010，$7 \odot 10$ 的8位二进制即为 00000101，也就是十进制的 5。
- 样例2：14 的8位二进制为 00001110，19 的8位二进制为 00010011，$14 \odot 19$ 的8位二进制即为 00001100，也就是十进制的 12。

Hint

可以先对 b 按位取反，然后再和 a 进行按位与运算。思考一下为什么可以这样做?

B 置0置1

题目描述:

给定一个 `unsigned int` 范围内的自然数 n 。已知 `unsigned int` 的二进制由 32 位 0 和 1 构成，记为 $s_{31}s_{30}...s_1s_0$ 。

现在，你需要对自然数 n 连续进行 t 次操作。每次操作都需要选择一个位置 k 进行操作，以及选择操作类型， 0 表示将第 k 位 s_k 设置为 0 ， 1 表示将第 k 位 s_k 设置为 1 。

输入

- 第一行包含一个整数 n ，表示初始整数。保证 n 为 `unsigned int` 范围内的整数。
- 第二行包含一个整数 t ，表示操作次数。保证 $0 \leq t \leq 1000$
- 接下来 t 行，每行包含两个整数，分别为位数 k 和操作类型 op （ 0 或 1 ），表示每次操作的位置和类型。其中保证 $0 \leq k \leq 31$ 。

输出

- 先输出 t 行，每行一个整数，表示每次操作后的结果。
- 最后一行输出一个整数，表示最终的结果。

输入样例

```
10
3
5 0
4 1
3 0
```

输出样例

```
10
26
18
18
```

样例解释:

初始整数是 10（二进制表示为 001010 ）。根据操作指令，分别执行以下操作:

1. 将第 5 位设置为 0 ，结果为 001010（二进制表示）。
2. 将第 4 位设置为 1 ，结果为 011010（二进制表示）。
3. 将第 3 位设置为 0 ，结果为 010010（二进制表示）。

最终的结果是 010010（十进制表示为 18 ）。

Hint

可以参考课件/书上例题3-5和3-6

简单的位运算哦~

C 高低位对调

题目描述

给定一组 `unsigned int` 范围的整数，已知 `unsigned int` 的二进制由 32 位 0 和 1 构成，记为 $s_{31}s_{30}\dots s_1s_0$。

请输出每个整数的二进制低 16 位和高 16 位对调后的数，即将整数的二进制码从 $s_{31}s_{30}\dots s_1s_0$ 变成 $s_{15}s_{14}\dots s_0s_{31}s_{30}\dots s_{16}$ 。

输入

共 $n+1$ 行。

第一行，一个正整数 n ，保证 $n \le 100$

接下来的 n 行，每行一个 `unsigned int` 范围的整数。

输出

输出 n 行

每行输出一个整数，表示将输入的整数低 16 位和高 16 位对调后的数。

输入样例

```
2
5
9
```

输出样例

```
327680
589824
```

样例解释

5的 32 位原码为 `00000000 00000000 00000000 00000101`

高低位对调后，得到 `00000000 00000101 00000000 00000000`,对应 327680

Hint

用 `scanf` 和 `printf` 读入和输出 `unsigned int` 类型的整数时格式化字符串要用 `"%u"` 哦~

Author：pyh

D 身份证号验证

题目介绍

中华人民共和国公民的身份证号码由 18 位数字或 X 组成，其中最后一位可能是 X。

身份证号码的前 6 位表示行政区划代码，第 7 位到第 14 位表示出生日期，第 15 位到第 17 位表示顺序码，第 18 位表示校验码。

现给定若干个身份证号，请检验身份证号是否合法。如果合法，输出 `YES`，否则输出 `NO`。

保证前 17 位数字合法，因此你只需要检验第 18 位校验码是否合法即可。

校验码的计算方法如下：

- 将前面的身份证号码 17 位数分别乘以不同的系数。从第 1 位到第 17 位的系数分别为 `7`，`9`，`10`，`5`，`8`，`4`，`2`，`1`，`6`，`3`，`7`，`9`，`10`，`5`，`8`，`4`，`2`。
- 将这 17 位数字和系数相乘的结果相加。
- 用加出来的和除以 11，看余数是多少。
- 余数只可能有 $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$ 这 11 个数。其分别对应的最后一位身份证的号码为 `1`，`0`，`X`，`9`，`8`，`7`，`6`，`5`，`4`，`3`，`2`。(即余数 0 对应 `1`，余数 1 对应 `0`，余数 2 对应 `X` ...)

输入

共 $n+1$ 行。

第一行一个正整数 n，保证 $1 \leq n \leq 50$。

接下来 n 行，每一行为一个身份证号。（若最后一位为 X，则为大写字母 `X`）

输出

输出 n 行。

每行表示身份证号码是否合法。如果合法，输出 `YES`，否则输出 `NO`。

输入样例

```
2
371311200312247819
130631197601191234
```

输出样例

```
YES
NO
```

Hint

在计算系数相乘结果之和时，除了直接写出表达式以外，我们也可以采用 “数组+循环” 的方式。

假设需要计算 $3\times5+9\times7+4\times9$

我们可以直接写 `sum=3*5+9*7+4*9;`

同时，我们也可以写成

```
int a[3] = {3,9,4};
int b[3] = {5,7,9};
int sum = 0;
for (int i = 0; i < 3; ++i)
{
   sum += (a[i] * b[i]);
}
```

看似代码量变大了，但如果需要计算 17 个系数相乘结果之和时，“数组+循环”的方法或许会更便捷且不易出错。

Author：pyh

E 质数、异或和一

题目描述

虽然 1 并不是一个质数，但是质数却能通过一些运算得到 1

接下来进行若干次询问 (询问次数不多于 10^5 次)，每次询问会给出两个自然数 m 和 n，请你判断从小到大第 m 和第 n 个质数进行**异或**运算的结果是否为 1，即判断 $p_m \oplus p_n = 1$ 是否成立。

输入格式

多组数据输入

每一组数据为一行，两个由空格分开的自然数 m, n，保证 $1 \leq m$, $n \leq 10^9$

输出格式

多组数据输出

对于每一组数据，输出一行：

- 若 $p_m \oplus p_n = 1$ 成立，输出 `YEEEEE!!!`
- 若 $p_m \oplus p_n = 1$ 不成立，输出 `WOOOOO!!!`

输入样例

```
8196 2168
```

输出样例

```
wOOOOO!!!
```

Author:星辰的微光

F 式神们夜里不睡觉 mini

题目背景

Yukari 姐姐彻底昏睡了，Ran 担心题目太难于是把题目改简单了一些！

题目描述

实现十以内任意进制的转换。

给出自然数的 M 进制表示，求它的 N 进制表示。

输入

多组数据输入。

第一行一个整数 T，表示数据组数。

接下来 T 行，每行两个整数 M, N，表示转换前后的进制基数；一个由数字构成的字符串，表示转换前的 M 进制自然数。输入均以一个空格分隔。

输出

每组数据一行，输出转换后的 N 进制整数。

输入样例

```
3
9 4 17251
6 7 523
9 9 142006884
```

输出样例

```
2321200
366
142006884
```

样例解释

对于第二组数据：

$$(523)_6 = 5 \times 6^2 + 2 \times 6^1 + 3 \times 6^0 = 3 \times 7^2 + 6 \times 7^1 + 6 \times 7^0 = (366)_7$$

数据范围

$1 \leq T \leq 100$，$2 \leq M, N \leq 10$，保证待转换的自然数在 `int` 范围内。

HINT

可以使用下面的方法输入字符串：

```
#include <stdio.h>
#include <string.h>
int main()
{
    char s[100]; //声明字符数组来存储字符串
    //...
    scanf("%s", s);
    int len = strlen(s); //存储字符串长度
    //...
    return 0;
}
```

如果此时读入的字符串是 `520` ，则 `len` 为 3， `s[0], s[1], s[2]` 分别为 `'5', '2', '0'` 。

G Te：离间计

题目描述

为了打败GYCY大军，DeNeRATe决定从其大军内部瓦解他，使出离间计，可以使GYCY大军中的士兵随DeNeRATe的意志进行策反（以下用D代表DeNeRATe，G代表GYCY）

士兵的编号从 $1 \sim n$，DeNeRATe会依次给出 m 条命令

- 当命令为 `1` 时，将G的所有士兵变为D的士兵，D的所有士兵变为G的士兵
- 当命令为 `2` 时，将编号最小的G的士兵策反为D的士兵（如果没有G的士兵了，则所有士兵都不发生改变）

  初始状态下，所有士兵都是G的，DeNeRATe想知道，最后会有哪些士兵是D的，以及总共多少个。

输入

第一行两个正整数 n, m，保证 $0 < n \leq 64, m \leq 10^6$，分别表示GYCY大军人数和DeNeRATe给出的命令条数

接下来 m 行，每行一个正整数 `1` 或 `2` 表示给出的命令

输出

共两行，第一行从小到大输出D的士兵的编号，用空格间隔

第二行，一个非负整数，表示最终有多少士兵是D的

输入样例

```
4 4
2
2
1
2
```

输出样例

```
1 3 4
3
```

Hint

可以用一个 `unsigned long long` 类型的数的存储当前士兵状态。每位上是0还是1表示这个位置上是G的士兵还是D的士兵。思考两个命令可以如何使用位运算实现?

对于样例而言，一共有编号为 $\{1,2,3,4\}$ 共 4 名士兵，最初都是G的士兵，可以用 `0000` 表示，每次命令后，变化如下:

1. 命令 `2` ： `0001`
2. 命令 `2` ： `0011`
3. 命令 `1` ： `1100`
4. 命令 `2` ： `1101`

因此最终编号为 $\{1,3,4\}$ 的士兵是D的士兵。

H 非标准进制

请注意本题时间限制与空间限制

题目描述

对于自然数 n 与大于 1 的正整数 $b_1,\cdots,b_m$，存在自然数 $a_0,a_1,\cdots,a_m$ 使得:

$$n=a_0+a_1b_1+a_2b_1b_2+a_3b_1b_2b_3+\cdots+a_mb_1b_2\cdots b_m$$

我们希望最小化 $a_0+a_1+\cdots+a_m$ 。

输入

第一行，`unsigned long long int` 范围内的自然数 n 。

第二行，用一个空格分隔的大于 1 的 `unsigned long long int` 范围内的自然数 $b_1,\cdots,b_m$，保证 $1\leqslant m\leqslant 10^7$。

输出

$a_0+a_1+\cdots+a_m$ 的最小值。

输入样例

```
95696
60 60 24
```

输出样例

```
93
```

提示

如果你超出时间限制（TLE），请思考你是否过多进行了不必要的处理。

如果你超出空间限制（MLE），请思考你是否过多使用了不必要的变量。

在 $a_0 + 60a_1 + (60 \times 60)a_2 + (60 \times 60 \times 24)a_3 = 95696$ 的条件下 $a_0 + a_1 + a_2 + a_3$ 的最小值是 93 。

注意到，1 分钟是 60 秒，1 小时是 60 分钟，1 天是 24 小时，所以 95696 秒是 1 天 2 小时 34 分钟 56 秒，从而 $56 + 34 + 2 + 1 = 93$ 。

其实，我们平常用的计时就相当于是混合进制的，60s = 1min，60min = 1h，24h = 1day 。

本题需要同学们对进制有深入的理解，类比常规的进制转换解决问题吧～

I 最小异或和

题目描述

给出 n 个正整数 $a_1, a_2, \cdots, a_n$ ，用正整数 x 与它们分别进行按位异或后再加和，求 x 使得和式最小，输出这个最小和。

即求 $\min\left\{\sum_{i=1}^{n}(x \oplus a_i) | x \in \mathbb{N}\right\}$ ，其中 $\oplus$ 代表按位异或。

输入

共两行。

第一行一个正整数 n ，保证 $1 \leq n \leq 10^6$ ；

第二行 n 个正整数 $a_1, a_2, \cdots, a_n$ ，保证均在 `unsigned int` 范围内。

输出

一行一个正整数，表示最小和。

输入样例

```
3
1 2 2
```

输出样例

```
3
```

样例解释

$x=2$ 时和式最小，$(2\oplus 1)+(2\oplus 2)+(2\oplus 2)=3$，故输出 3。

Hint

异或运算对于每一位是独立的。

Author: $\mathcal{D}avid\ \mathcal{T}ong$, aka 哪吒

J 导弹密码

题目描述

邪恶的DeNeRATe为达到他不可告人的秘密，向某地区发射了一枚物质泯灭导弹，为阻止DeNeRATe的阴谋，你向先知GYCY求助，GYCY告诉你，这枚导弹内部有一个长度为 n 的自然数序列 $a_1,a_2,\cdots,a_n$，你需要将其划分为 m 段连续的区间，使得 $passworld$ 值**最小**，$passworld$ 即为**终止导弹的密码**。$passworld$ 定义如下：

$$passworld=(a_1\oplus a_2\oplus\cdots\oplus a_{x_1})\vee(a_{x_1+1}\oplus a_{x_1+2}\oplus\cdots\oplus a_{x_2})\vee\\ \cdots\vee(a_{x_{m-1}+1}\oplus a_{x_{m-1}+2}\oplus\cdots\oplus a_n)$$

其中，符号 $\oplus,\vee$ 分别表示按位异或和按位或运算

输入

第一行，两个正整数 n,m，保证 $1\le m\le n\le 5\times 10^5$

第二行，n 个自然数 $a_1,a_2,\cdots,a_n$，保证均在 `unsigned int` 范围内

输出

一行，表示导弹的终止密码

输入样例

```
5 3
9 5 2 7 1
```

输出样例

```
9
```

- End -