E2-Statement

- 练习时间2022/03/19 08:00 - 2022/03/22 23:50
- 总共10道题，编号为A~J，题目不一定按难度排序，同学们可以勤看榜单，按照榜单通过人数答题；或者提前读完所有题目，按照自己擅长题目的情况答题。
- 各题分值分布为如下，总分101分，满分100分（超过100分按100分计算）：

序号	A	B	C	D	E	F	G	H	I	J
分值	20	20	20	10	10	10	3	3	3	2

- 请严肃练习，严禁抄袭他人代码，课程组会在练习结束后进行代码查重并给予警告。

组题：cbd、爱吃猪脚的猪脚、ljf

A 简单浮点数(水题速来)

题目描述

为了迎接开学，北航优购所有商品打95折！

小L兴冲冲跑去优购买了一堆东西，请你帮他计算最后所有商品应付的价格吧！

输入格式

第一行，一个整数 n ，表示小L一共买了 n 件商品；

接下来 n 行，每行一个浮点数 a_i，代表每一件商品的价格，这里的价格是原价。

输出格式

一行，一个浮点数，表示实际付款，四舍五入保留两位小数。

输入样例

```
2
5.00
6.00
```

输出样例

```
10.45
```

数据范围

$0 \leq n \leq 10$

$0 < a_i < 10$

B 简单位运算

题目描述

计算三个有符号整数按位异或的值:A xor B xor C (其中xor表示按位异或)

输入

一行3个有符号整数A,B,C(保证在long long范围内)

输出

对于每组数据，输出一行， A xor B xor C的值

输入样例1

```
1 1 1
```

输出样例1

```
1
```

输入样例2

```
1 1 -1
```

输出样例2

```
-1
```

提示

点击获取按位异或运算的说明

按位异或用`^`

```c
#include <stdio.h>
int main() {
    int a = 3, b = 4;
    printf("%d", a ^ b);
    return 0;
}
```

author:cbj

C 简单的字符串统计

题目描述

输入一个字符串s，输出该字符串的长度n

输入

一行非空的字符串s

输出

该字符串的长度n

输入样例1

```
Hello,world!
```

输出样例1

```
12
```

输入样例2

```
hello world
```

输出样例2

```
11
```

Hint

输入行末没有换行符，不要使用换行符作为判断输入结束的依据。

数据范围

$0 < n \leq 500$

Author: rjy

D 2的次幂

题目描述

判断一个正整数 n 是不是 2 的次幂 $(2^0, 2^1, 2^2, \cdots)$

输入格式

一行，一个正整数 n

输出格式

输出共两行；

第一行，如果 n 是 2 的次幂则输出 `Yes` ，否则输出 `No` ；

第二行，输出不超过 n 的最大的 2 的次幂；

输入样例1

```
4
```

输出样例1

```
Yes
4
```

输入样例2

```
17
```

输出样例2

```
No
16
```

数据范围

$0 < n \leq 2^{32} - 1$

HINT

考虑一下位运算的做法

如果你做了水水的a+b>c，你应该注意一下数据范围，unsigned int 的数据范围是 $[0, 2^{32} - 1]$，占位符是 `%u`

E 找内鬼

题目描述

据比较可靠的线报，`2021级-航空航天大类-C语言程序设计`课程内混进了一个内鬼！

内鬼非常狡猾，将自己的编号伪装成了一位本来存在于课程内的同学的编号。这样，拥有点名册的助教们也无法依靠一一对照来找出内鬼。同时由于课程人数有五万甚至四万人之多，助教们也很难靠自己一个一个去找里面重复的编号。正当大家一筹莫展之际，czy突然站出来说："太简单啦，咱们只要**有课程内的编号和点名册上的编号**，一定能找出内鬼！"

助教们把课程内的编号和点名册上的编号交给了czy，但其实czy也不知道这样做有什么用，于是他把这堆数据给了你，请你来帮帮他。

简单描述

在输入的 n 个数中**至多**有1个数出现了奇数次，其他数字都出现了偶数次，请输出这个数；如果所有数字都出现了偶数次，输出`False Alarm.`

输入

共2行。

第一个数为输入数据的总数 n。

接下来一行共有 n 个整数，每一个整数以空格隔开，代表课程内的编号或点名册上的编号。由于课程内有老师同学和教职工等，因此编号的位数并不统一。

因为助教们输入编号时手忙脚乱，输入时并不按照 "课程内编号——点名册" 的顺序输入。

输出

输出一行整数，表示内鬼伪装成的编号；若线报有误，实际上没有内鬼的话， 输出`False Alarm.`。

保证内鬼最多只会有一个。

输入样例1

```
7
2345 34567 456789 34567 456789 2345 456789
```

输出样例1

```
456789
```

输入样例2

```
4
1234 2345 1234 2345
```

输出样例2

```
False Alarm.
```

数据范围

保证输入数据总数n满足$1 \leq n < 10^6$，对每一个编号a_i，有$1 \leq a_i < 10^9$。

HINT

有没有一种可能，当然我只是说可能，这道题可以用异或运算去解决

关于异或运算的一些性质（以下 $\bigoplus$ 为异或运算）：

- $0 \bigoplus a = a$
- $a \bigoplus a = 0$
- 异或具有交换律
- 异或具有结合律

Author: czy

F LJF赶校车

题目描述

众所周知，航C助教其实大部分都住在学院路。某天上机结束后，LJF发现只剩下最后一班校车了，但是点评还没有开始，数学不好的LJF需要大家来帮忙计算能搭上校车的概率是多少。

根据同学们答疑的热情程度，LJF的下班时间是不固定的，但可以保证在`h1:m1`到`h2:m2`之间到达车站；同时校车司机也有睡过头的可能，但同样保证校车在`h3:m3`到`h4:m4`之间到达车站，且每时刻到达车站的概率相等。有且仅有LJF比校车早到车站才能搭上校车，求成功搭上校车的概率。

输入

共2行数据。

第1行为`h1:m1 h2:m2`。

第2行为`h3:m3 h4:m4`。

格式均为`hh:mm`，为相同时区的同一天的24小时制的时刻。

保证`h1:m1`在`h2:m2`之前，`h3:m3`在`h4:m4`之前，数据均为非负整数。

输出

如果LJF一定能搭上校车，输出`GOOD!`。

如果LJF一定搭不上校车，输出`0HNO!`。

其余情况输出一个8位小数(四舍五入)，表示LJF能搭上校车的概率。

输入样例1

```
20:30 21:00
21:30 22:00
```

输出样例1

```
GOOD!
```

输入样例2

```
20:00 21:30
20:30 22:00
```

输出样例2

```
0.77777778
```

数据范围

$$0 \leq h1, h2, h3, h4 \leq 23$$

$$0 \leq m1, m2, m3, m4 \leq 59$$

HINT

高中概率几何概型模型，忘记了点这个和相关题目参考

PS

当然LJF一般都会不要脸的蹭老师车回学院路，搭不上车是不可能哒！！

G 今天也要牵绊变身

题目描述

牵绊变身，是呱呱泡蛙进化后甲贺忍蛙的专属特性。通过和训练家交织重叠的想法进行同步，即可切换成超帅的形态。

呱呱泡蛙今天也想和它的训练家小智完成牵绊变身。但是同步两人的想法需要解密码。解密码和如下加密算法有关。对于无符号整数 n ，记一个加密映射 f ：

$$f(n) = n \oplus (n << 1)$$

这里的记号 $\oplus$ 表示按位异或，在C语言中是键盘6上方的尖角。该算法为，将 n 与 n 左移一位进行异或。数学上可以证明，这个加密方式是一一对应，对于每个 n 存在唯一的 $f(n)$ ，对于每个 $f(n)$ 也存在唯一的 n 。

小智给呱呱泡蛙发来他的密文，即 $f(n)$ ，呱呱泡蛙需要反过来求解明文 n ，明文匹配，才能完成牵绊变身。

好在聪慧的呱呱泡蛙在**本页面的最下方**提前告诉了你解码原理。根据课上的位运算知识，你可以实现它！

（注：对于unsigned int类型的输入输出，可以使用%u。）

输入

多组输入，每行一个无符号整数，是加密后 $f(n)$ 的值。

输出

多组输出，每行一个无符号整数，是加密前 n 的值。

输入样例

```
656
657
658
```

输出样例

```
4294966896
399
398
```

样例解释

对于unsigned int类型的4294966896、399、398，作用一次映射 f 之后，会得到656、657、658。

Hint

呱呱泡蛙给出的解码原理大致按照如下思路。

虽然我们不知道 n ，但是加密后的值 $f(n)$ 是已知的。

为避免混淆，记 $m = f(n) = n \oplus (n << 1)$ ，已知的是加密后的值 m 。

对已知的 m 作用映射 f ：

$$f(m) = (n \oplus (n << 1)) \oplus ((n \oplus (n << 1)) << 1)$$

$$= n \oplus (n << 1) \oplus (n << 1) \oplus (n << 2)$$

$$= n \oplus (n << 2)$$

因此对于已知的加密后的值 m ， $f(m)$ 是 n 和 n 左移 2 位进行异或的值。假如记一个新映射 g ：

$$g(n) = n \oplus (n << 2)$$

对上述 $f(m)$ 作用映射 g ：

$$g(f(m)) = (n \oplus (n << 2)) \oplus ((n \oplus (n << 2)) << 2)$$

$$= n \oplus (n << 2) \oplus (n << 2) \oplus (n << 4)$$

$$= n \oplus (n << 4)$$

因此对于已知的加密后的值 m ， $g(f(m))$ 是 n 和 n 左移 4 位进行异或的值。假如记一个新映射 h ：

$$h(n) = n \oplus (n << 4)$$

对上述 $g(f(m))$ 作用映射 h ：

$$h(g(f(m))) = (n \oplus (n << 4)) \oplus ((n \oplus (n << 4)) << 4)$$

$$= n \oplus (n << 4) \oplus (n << 4) \oplus (n << 8)$$

$$= n \oplus (n << 8)$$

因此对于已知的加密后的值 m ， $h(g(f(m)))$ 是 n 和 n 左移 8 位进行异或的值。以此类推。

一个事实是，n 在计算机中存储是**有限长**的，这里的“长度”指类型长度，包含 n 最高位左端的若干 0 。上述操作若干次之后，得到 n 和 n 左移 k 位进行异或的值，此时**左移位数 k 已经超过了 n 的类型长度**，相当于 n 左移 k 位的结果变为 0 ，于是 n 与 0 的异或还是 n ，最终求得 n 的值，整个解码过程即宣告结束。

问题的关键变为：对于unsigned int类型的 n ，究竟有多长？即，上述操作要执行到什么时候停止？呱呱泡蛙觉得这个问题在课上已经讲过，留给你自行思考吧！

题目原型：编号881，JumbledCommunication。

Author：未来拥有牵绊变身特性的呱呱泡蛙

H 5421码

题目描述

BCD码，是一种二进制的数字编码形式，用4位二进制数来表示1位十进制数中的$0-9$这10个数码。这种编码形式利用了四个位元来储存一个十进制的数，使二进制和十进制之间的转换得以快捷的进行。相对于一般的浮点式记数法，采用BCD码，既可保存数值的精确度，又可免去使计算机作浮点运算时所耗费的时间。此外，对于其他需要高精确度的计算，BCD编码亦很常用。

BCD码可分为有权码和无权码两大类，有权BCD码有8421码、2421码、5421码等，无权BCD码有余3码，格雷码等，本题所讨论的5421码为有权码。

有权是指四位二进制数中每一位数码都有确定的位仅值，若把这四位二进制码按权展开，就可求得该二进制码所代表的十进制数。例如5421码，从左至右每位的权分别是十进制5、4、2、1。对于1011这个代码按权展开就是$1\times5+0\times4+1\times2+1\times1=[8]_{10}$，可见$5421$码中的代码$1011$代表了十进制数$8$。

5421码的编码方式不是唯一的，有的十进制数码存在两种加权方法，如数码5既可以使用1000表示，也可以用0101表示，为避免歧义，将本题所采用的编码方案规定如下，除此十个代码以外，余下的六个代码规定为非法码，非法码不允许出现，若出现则识别为错误。

十进制数	5421BCD码	十进制数	5421BCD码
0	0000	5	1000
1	0001	6	1001
2	0010	7	1010
3	0011	8	1011
4	0100	9	1100

对于任意一个十进制正整数a，都有唯一确定的5421码与之对应，比如与197相对应的的5421码为$0001,1100,1010$，该编码中$0-3$位代表十进制的7，而$4-7$位代表十进制的9, 而$8-11$位代表十进制的1，十进制的每一位与5421码的每四位相对应。

注意到BCD码与二进制码一样，都是0与1的序列。若将此BCD码视作普通的二进制代码，又有一个十进制正整数b与此代码对应。如将197的5421码看作普通二进制，则有十进制458与之对应。

小冰学习了二进制码和BCD码的基本知识，想探究上述内容中十进制正整数a与b的差异，即a与b的二进制码中不同位的个数。如197的二进制为011000101，458的二进制为111001010，不同位的个数为5。

输入

一行，一个十进制正整数a

输出

一行，两个十进制正整数b, c,十进制正整数b的二进制码恰为十进制正整数a的5421码，c为十进制正整数a与b的二进制码不同位的个数

输入样例

```
197
```

输出样例

```
458 5
```

数据范围

$0 < a < 10^{17}$

Hint

1.请仔细观察$0-9$的$5421BCD$码与其二进制码之间的某种对应关系，注意$0-4$与$5-9$两组数有着不同的对应关系。

2.表示一个相同的十进制数，和二进制码相比，BCD码往往需要更多位，因此请再三斟酌数据范围。

3.注意到异或运算使两位相同为0，不同为1，因此可以利用异或运算来求解a和b的二进制不同位的个数。

Author:LNB

I 浮点数协议

题目描述

在IEEE 754标准中，一个 32 位的二进制单精度浮点数（`float`类型数据）通常由如下形式来表示：

$se_{k-1}\ldots\ldots e_1e_0f_{n-1}\ldots\ldots f_1f_0$ （其中$k=8, n=23; s, e_i, f_j \in \{0,1\}$）

IEEE 754标准规定，通过如下公式计算一个有效的、非无穷大的二进制浮点数对应的十进制实数值 x：

$$x = (-1)^s \times F \times 2^{E-bias}$$

其中，F 是一个二进制小数，E 和 $bias$ 是十进制整数，$bias = 2^{k-1}-1 = 127$，计算 F 和 E 需要分情况讨论：

- 本题不涉及 $e_{k-1}\ldots\ldots e_1e_0$ 的每一位全为 1的情况；
- 若 $e_{k-1}\ldots\ldots e_1e_0$ 的每一位全为 0 ，则 $F = 0.f_{n-1}\ldots\ldots f_1f_0, E = 1$；
- 若 $e_{k-1}\ldots\ldots e_1e_0$ 的每一位既不全为 1 也不全为 0 ，则 $F = 1.f_{n-1}\ldots\ldots f_1f_0$，$E$为二进制数 $e_{k-1}\ldots\ldots e_1e_0$ 转换为十进制数后得到的整数值。

给予一个`float`类型的数据，请你提取出它的 F 和 E（F 用二进制小数表示，E 用十进制整数表示）

输入

多组数据输入，每行一个`float`类型的浮点数。

数据组数不超过10^5行。

输出

对于每组数据输出一行：它的 F 和 E，用空格隔开。

输入样例

```
12.123
-456.78
0.00000000000000000000000000000000000000001
```

输出样例

```
1.10000011111011111001111 130
1.11001000110001111010111 135
0.00000000000001011001010 1
```

样例说明

输入：12.123

- 实际在计算机中的二进制表示：$0\ 10000010\ 10000011111011111001111$
- 由于10000010不全为0：
  - E为10000010转换为10进制：130
  - F为：$1.10000011111011111001111$

输入：-456.78

- 实际在计算机中的二进制表示：$1\ 10000111\ 11001000110001111010111$
- 由于10000111不全为0：
  - E为10000111转换为10进制：135
  - F为：$1.11001000110001111010111$

输入：$0.00000000000000000000000000000000000000001$

- 实际在计算机中的二进制表示：$0\ 00000000\ 00000000000001011001010$
- 由于00000000全为0：
  - E为：1
  - F为：$0.00000000000001011001010$

HINT

使用`<string.h>`函数库中的`memcpy`函数可以将一个`float`类型变量逐比特地复制进一个`int`类型变量，具体用法可以参考如下代码：

```
#include <stdio.h>
#include <string.h>

int main() {

    float a = 3.14;
    int da;
    memcpy(&da, &a, 4); // 将float类型变量a逐比特复制进int类型变量da

    return 0;
}
```

Author: Lucien Li

J n重循环移位

题目描述

给予一个有n个`int`型数据的数组`a`，在C语言原有的左移以及右移运算的基础上，定义`a`的一次**向左（右）的**“n重循环移位”操作为：

对于每一个数组中的元素`a[i]`：$(0 \leq i < n)$

- 第一步：
  - 对于**向左的**“n重循环移位”：将`a[i]`**左移**一位，溢出的位记为`b[i]`；
  - 对于**向右的**“n重循环移位”：将`a[i]`**右移**一位，溢出的位记为`b[i]`；
- 第二步：
  - 对于**向左的**“n重循环移位”：将`a[i]`的**最低位**设置为`b[i + 1]`（若`i + 1`等于n，则设置为`b[0]`）；
  - 对于**向右的**“n重循环移位”：将`a[i]`的**最高位**设置为`b[i - 1]`（若`i - 1`等于-1，则设置为`b[n - 1]`）。

（HINT中给出了具体的示例）

现在给予你一个有n个`int`型数据的数组`a`，请你输出经过t次向左或向右的“n重循环移位”后的数组`a`。

输入

多组数据输入。

每组数据有三行：

第一行为一个正整数n，代表数组中的元素个数；

第二行为n个整数，依次代表数组`a`中的所有元素，每个数据之间用一个空格隔开；

第三行有：一个字母，只可能是`l`或`r`，`l`代表向左进行操作，`r`代表向右进行操作；以及一个正整数t，代表执行操作的次数。每个数据之间用一个空格隔开。

其中数组`a`的中的数据大小不会超过`int`类型的数据范围，$0 < n \leq 10^5, 0 \leq t < 2^{32}$.

数据组数不超过10^3组。

输出

对于每组数据，输出一行：依次输出经过t次"双重循环移位"操作的数组`a`中的所有元素，每个数据之间用一个空格隔开。

输入样例

```
5
1 2 3 4 5
l 109
5
1 2 3 4 5
r 345
```

输出样例

```
32768 40960 8192 16384 24576
640 128 256 384 512
```

HINT

以有三个8位二进制数的数组举例说明一次向左的"n重循环移位"：

```
.------------>----------->------------.
|     a[0]       a[1]       a[2]     |  ===>    a[0]       a[1]       a[2]
'<--10000000<--11111101<--00001000<--'        00000001   11111011   00010001
    ^          ^          ^
```

以有三个8位二进制数的数组举例说明一次向右的"n重循环移位"：

```
.------------<-----------<------------.
|     a[0]       a[1]       a[2]     |  ===>    a[0]       a[1]       a[2]
'-->10000001-->01111111-->00001000-->'        01000000   10111111   10000100
           ^          ^          ^
```

Author: Lucien Li