C6-2021级航类-第六次上机

A 简单的统计

题目介绍

期中考试结束咯！大家的成绩总体来讲都很不错呀。

JJJ 这里有 n 个同学的成绩，现在想请你统计，分数在 $[60,80)$ 这个区间的同学的分数平均值。

输入格式

第一行，一个正整数 n ，表示学生的个数。

第二行，一共 n 个整数，分别表示 n 个同学的成绩。

输出格式

一个小数，表示分数在 $[60,80)$ 这个区间的同学的分数平均值。结果保留**2位小数**。

输入样例1

```
10
65 79 70 86 66 68 85 90 51 55
```

输出样例1

```
69.60
```

数据范围

对于 100% 的数据：$1 \leq n \leq 10^6$, 分数为 $[0,100]$ 之内的整数.

Author：///

B LJF排排坐

题目描述

大家都知道 LJF 赶不上校车，但是都不知道为什么赶不上，其实是因为 LJF 有拖延症！！

这天老师让 LJF 根据期中考试成绩给大家排个序，但 LJF 实在是动不起来，只能由你来帮 LJF 完成任务了。

输入

两行。

第一行为一个正整数 n $(n \leq 2^{20} = (1 << 20) = 1048576)$ 。

第二行为 n 个整数 $a_i (a_i \in [-2147483648, 2147483647])$ ，以空格隔开。

输出

一行。

将 n 个整数由小到大输出，以空格隔开。

输入样例

```
5
4 3 7 2 9
```

输出样例

```
2 3 4 7 9
```

Hint

由于需要排序的数字比较多，如果使用时间复杂度为 $o(n^2)$ 的排序方法最多只能拿到 0.6 ，会 TLE 哒！！

如果 TLE 了，思考一下课上有没有讲过某种更快速的排序方法

C 爱祖国、爱人民、爱劳动、爱科学、爱社会主义

题目描述

“爱祖国、爱人民、爱劳动、爱科学、爱社会主义”是社会主义道德建设的基本要求。请判断字符串中是否包含上述五项要求之一。

输入

共两行。

第一行，一个字符串 R，
$R \in [Love\ Motherland, Love\ People, Love\ Labor, Love\ Science, Love\ Socialism]$ 。

第二行，一个字符串 S，仅包含大小写字母、数字和空格，长度不超过 2000 。

输出

共一行。

如果 S 中包含 R ，输出一个数字 x ，表示**首次出现**的 R 的第一个字母是 S 的第 x 个字符（从 1 开始计数）。

如果 S 中不包含 R ，输出 `Oh!No!` 。

本题区分大小写。

输入样例

```
Love People
Love Motherland Love People Love Labor Love Science Love Socialism
```

输出样例

```
17
```

Hint

听说 `string.h` 中有这个函数：

```
strstr(const char *_Str,const char *_SubStr)  // 该函数返回_Str中指向_SubStr首次出现位置的指针，没有_SubStr则返回NULL
```

Author：爱吃猪脚的猪脚

D Laika的密码

题目描述

2065年，探测器 `大眼睛猫猫一号机` 在冥王星轨道上检测到了来自 `Laika` 的信号。`大眼睛猫猫一号机` 与 `Laika` 建立通信，为了保证信息传输的安全性，`Laika` 对每条信息（明文）采取了加密，并将密文发送给 `大眼睛猫猫一号机` 。解密规则如下:

1.当密文为回文字符串时，明文为密文所有偶数位置上的字符按顺序组成的新字符串

2.当密文不为回文字符串时，明文为密文所有奇数位置上的字符按顺序组成的新字符串

请你帮助探测器完成解密。

输入格式

一行密文字符串。

输出格式

一行明文字符串。

输入样例1

```
laaaiakaaasataialalawaaananaaagaoahaoamaea
```

输出样例1

```
laikastillwannagohome
```

输入样例2

```
VFYFKZLHNKKNHLZKFYFV
```

输出样例2

```
FFZHKNLKYV
```

数据范围

保证密文长度不超过 10000，只包含 26 个字母的大小写，无不可见字符。

HINT

`回文串` 是一个正读和反读都一样的字符串，例如 `level` 和 `anna` 都是 `回文串` 。

PS

`лайка` 是第一个上太空的地球生物，俄语名称是“吠叫着的生物”。她是一只苏联太空狗，是借由苏联太空载具史波尼克二号的第一位踏上轨道的活乘客。她于 1957 年 11 月 3 日当地时间上午 10 时 28 分升空，但几小时后因太空衣隔热不佳死去。目前它与当年的太空舱还滞留在地球轨道上。

E 小小的统计

题目描述

输入若干个数，其中有整数也有小数，分别统计其中整数的个数和小数的个数，并分别求所有整数的平均值与所有小数的平均值。

输入

共一行，其中若干个数，两个数之间用空格隔开。

注意：数 `2.0` 被视为小数，我们可以认为整数与小数的区别仅在于它们有没有小数点。

数据保证所有的数字大于 -10^5 ，小于 10^5 ，且数字总数不超过 10^4 。

输出

共两行。

第一行，一个整数与一个浮点数，中间用空格隔开，分别代表整数的个数与整数的平均值。

第二行，一个整数与一个浮点数，中间用空格隔开，分别代表小数的个数与小数的平均值。

平均值保留五位小数。

输入样例

```
1 1.0 2 2.0 3 3.0 4.4
```

输出样例

```
3 2.00000
4 2.60000
```

样例解释

整数共 3 个，分别为 $1, 2, 3$ ，平均值为 2.00000 。

小数共 4 个，分别为 $1.0, 2.0, 3.0, 4.4$ ，平均值为 2.60000 。

HINT

1.可以使用 `string.h` 头文件中的 `strchr` 函数查找字符在字符串中首次出现的位置，如果找到则返回该位置的地址，如果未找到则返回 `NULL` ，用法如下：

```
char s[100],*p;
scanf("%s",s);
p=strchr(s,'a');
if(p==NULL)printf("字符串里面没有字符a");
else printf("字符串中首次出现的字符a的地址为%d",p);
```

2.可以使用 `stdio.h` 头文件中的 `sscanf` 函数从一个字符串中读进与指定格式相符的数据，用法如下：

```
char s[]={"123.456"};
double a;
sscanf(s,"%lf",&a);//其他数据类型同理，同时读进多个亦可。
printf("%.3lf",a);//输出为123.456
```

F 宋老师的名次预测4.0

题目描述

嘿嘿嘿嘿名次预测 4.0 来咯！预测这么好玩！为什么不玩！

活动火爆，这次参赛的人数太多太多了，有 2147483647 个，并且**依次获得了第 1 到第 2147483647 名。**宋老师不可能对所有参赛人数的名次进行预测，因此只预测了前 n_i 名，请你判断一下宋老师猜对的人数吧！

输入格式

多组数据输入；

每组数据一行，为 n_i 个空格分隔的正整数，为宋老师的一组预测；

每组的 n_i 可能不同，但不超过 100；

输出格式

对于每组数据输出一行，一个非负整数，为宋老师这组预测猜对的人数。

输入样例

```
1 2 3
2 1 3 1989
1 2 3 2147483647 520 1314 72629
```

输出样例

```
3
1
3
```

样例解释

对于样例输入的第二组数据，宋老师预测 2 号、1 号、3 号、1989 号选手依次获得前 4 名，显然只猜对了一个人，故输出 1；其他同理

HINT

每一行末尾保证没有空格，最后一行没有换行符

对，就是没有告诉你 n_i 具体是多少，试试 `sscanf` +指针的用法吧

AUTHOR：Songyou，cbd

G 图图的红包区间合并

题目描述

搞明白 vx 红包机制后，图图还是抢不到手气最佳，于是开启 emo 状态 (。•́‿•̀。)

为了搞清楚是手机的问题还是人品的问题，图图找来 n 部手机，每部手机抢若干次红包，记下所抢得的最小红包金额 min 和最大红包金额 max，于是该部手机所能抢得的金额区间为 $[min, max]$，现请你统计出这 n 部手机所能抢到的金额区间。

一句话题面：给出 n 个浮点数闭区间进行**区间合并**，将这些区间合并为不相交的闭区间。

输入格式

第一行，一个正整数 n，表示区间个数。

接下来 n 行，每行包含两个以空格分隔的浮点数 min、max，表示红包区间 $[min, max]$，每个浮点数**有且仅有两位小数**。

输出格式

输出若干行，表示合并后的区间，每行包含两个以空格分隔的浮点数 $left$、$right$，表示区间 $[left, right]$，各区间按升序排列输出。

输入样例

```
4
10.00 66.66
66.66 123.45
5.55 9.99
0.05 6.66
```

输出样例

```
0.05 9.99
10.00 123.45
```

数据范围

$0 < n < 2 \times 10^6$

$0 < min < max < 200$

HINT

- 数据量这么大，排序行不通的哦
- 对浮点数四舍五入取整请使用 `round` 函数
- 稍微改动课件上的代码即可：

散列表的应用实例—区间合并

```c
//example 6-13.c
#include <stdio.h>
#define MAX_N (1024*1024)
int range[MAX_N];

void print_zone()
{
    int i, n;
    for(i=0; i<MAX_N; i++){
        if(range[i]==0)   // 没有区间
            continue;
        printf("%d ", i);
        for(n=range[i]; i<=n; i++)
            if(range[i] > n) // 新区间更远
                n = range[i]; // 扩展右区间
        i--; // 回到当前区间①的上界n
        printf("%d\n", i);
    }
}
```

```c
int main(){
    int a, b;
    while(scanf("%d%d",&a, &b) == 2){
        if(b>range[a])
            range[a]=b;
    }
    print_zone();
    return 0;
}
```

初始化（相同下界时，扩展上界）

[1,4]
[1,5]
→ [1,5]

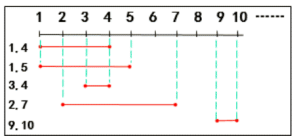
i为当前区间①的下界（左界）

区间有交叉时（i<=n ），上界（右界）扩展（if语句成立时上界扩展 n = range[i]；if不成立，新区间不存在或包括在区间①中）。

该部分代码完成，第一个区间的上界不断扩大，直到[1,7]。（注意：n是变化的 ）

打印扩展后的上界

即便输入顺序为：
9 10
2 7
3 4
...
区间[9 10]被hash到最后的位置，即9，是最后一个区间，无需排序。即，每个区间是一个元素，被hash后，hash的位置是该区间的左界。



题面灵感来源于 zym:爱偷懒的球场管理员 + lrq:计数排序

Author: wqh

H zhnの镜像串

题目描述

魔法少女zhn定义了一种镜像字符串：这个字符串由下面这些字母组成，如果这个字符串关于**正中间的位置完全镜像对称**，那么我们称他为好的串。

具体举几个栗子：

oHo 是一个好的串，因为 H 是轴对称的，o 与 o 也关于 H 轴对称。

aa 不是一个好的串，因为两个 a 关于中间不是镜像对称。

也就是说，如果我们字符串的长度是奇数，**最中间**的字母必须是轴对称的，如果我们字符串长度是偶数，那么两边的字母必须关于中间轴对称。

注意：m 和 n 两个字母不是轴对称的，i 也不是，b 与 d 对称，p 与 q 对称。具体是不是参考上面 zhn 给的那张图。

输入

多行字符串，保证只包括上面那些字母，字符串长度 $s <= 1000$ 。

输出

每组数据输出一行，如果是一个好串，输出 `"Cool"` ，否则，输出 `"OH,NO!"` (双引号不需要输出)。

输入样例

```
bod
```

输出样例

```
Cool
```

author:魔法少女zhn

I 小型文本编辑器

题目描述

请你实现一个小型文本编辑器。给予你一个字符串 `text` ，你需要实现如下操作：

- 查找：查找 `text` 中的所有特定字符串；
- 替换：将 `text` 中的所有特定字符串替换为另一个字符串；
- 删除：将 `text` 中的所有特定字符串删除；

注意在 `text` 中匹配特定字符串时，优先匹配先出现的，且在一次替换（删除）操作中新出现的字符串不应该参与到当前操作的匹配中。

输入

第一行是一个字符串，代表 `text` ；

接下来是多个操作的输入，代表对 `text` 的操作序列，每组数据包含：

- 一行整数，代表操作类型：1代表查找，2代表替换，3代表删除；
- 对于查找和删除操作，接下来还会有一行字符串 `str` ；
- 对于替换操作，接下来还会有两行字符串 `str` 和 `replace_str` 。

保证 `text` 、 `str` 和 `replace_str` 不为空且长度小于 10^3 ，并且保证 `text` 在操作过程中的长度总是不超过 10^3 。输入的字符串中只包含可见字符。

保证每一行最后都有换行符且使用 `\n` 进行换行。

输出

对于每个数据，输出一行：

- 若该组数据对应的操作是替换或删除，则输出经过操作后的 `text` ；
- 若该组数据对应的操作是查找，输出 `str` 在 `text` 的第几个字符位置出现（从1开始计数）。若没有出现，则输出 `NOT EXIST` ；若出现了多次，则升序输出所有出现的位置，用一个空格隔开。

输入样例1

```
I love Beijing University of Aeronautics and Astronautics.
1
au
3
 of Aeronautics and Astronautics
2
jing
hang
1
Peking University
```

输出样例1

```
35 52
I love Beijing University.
I love Beihang University.
NOT EXIST
```

输入样例2

```
BBAAABABA
1
AA
2
ABA
A
3
BA
2
AA
AD
```

输出样例2

```
3 4
BBAAABA
BAA
BAD
```

BAD是MJ的第三张专辑

Author: Lucien Li

J 安全小达人

题目描述

zym 发现自己的 QQ 已经三年没有改密码了！为了保证自己的账号安全，安全小达人 zym 修改了之前的密码。zym 想看看自己的新密码和旧密码的差别有多大，请你来帮她算一算。

定义新密码 $newpw$ 和旧密码 $oldpw$ 的差别为将**新密码** $newpw$ **转换成** $oldpw$ 所使用的最少操作数。

你可以对密码进行如下三种操作：

- 插入一个字符
- 删除一个字符
- 替换一个字符

输入格式

两行。

第一行为一个字符串，表示新密码 $newpw$ 。

第二行为一个字符串，表示旧密码 $oldpw$ 。

输出格式

一行。

第一行为一个整数，表示新密码和旧密码的差别。

输入样例

```
zym@!@12345666
zymm$$123666
```

输出样例

```
5
```

样例解释

`zym@!@12345666` -> `zymm!@12345666`（将 `@` 替换成 `m` ）

`zymm!@12345666` -> `zymm$@12345666`（将 `!` 替换成 `$` ）

`zymm$@12345666` -> `zymm$$12345666`（将 `@` 替换成 `$` ）

`zymm$@12345666` -> `zymm$$1235666`（删除 `4`）

`zymm$@1235666` -> `zymm$$123666`（删除 `5`）

数据范围

记 $|newpw|$ 表示 $newpw$ 的长度，$|oldpw|$ 表示 $oldpw$ 的长度

$1 \leq |newpw|, |oldpw| \leq 1000$

密码中可能包含大写字母 $A-Z$，小写字母 $a-z$，数字 $1-9$，以及特殊符号 `!,@,#,$,%` 。

HINTS

1. 为了账号安全偶尔换换密码也是不错的
2. 如果没有思路的话，可以先从**子问题**思考。（子问题就是比原问题规模更小，其余相同的问题）

$AUTHOR: zym$

K 终极进制

题目描述

在遥远的枝江边陲，有一群勇敢的牛牛民快乐地生活着。牛牛民是远近闻名的工匠，制作平底锅和锤子尤其拿手，由于声名远扬，很多人都想用金币购买牛牛民的平底锅，但枝江通用的是十进制，而牛牛民却习惯使用 `终极进制` 计数，该进制有一个口诀：

> 第一次勇敢牛牛
>
> 第二次勇敢牛牛
>
> 终极勇敢牛牛

这个口诀的意思是，牛牛民每数满 `2 + 1` 时，就会大喊一声 `moo~` 表示 `终极` ，每数满 `终极 + 1` 时，就会进一位。现在，请聪明的你帮笨笨的牛牛民将十进制的金币数量转换为 `终极进制` 的金币数量吧。

输入

输入包含多组测试数据。

每组测试数据占一行，包含一个**长度**不超过 1000 位的十进制非负整数，**该整数可能拥有前导零。**

输出

每组数据输出一个结果，占一行，为输入对应的 `终极进制` 数，该数不含前导零。

输入样例

```
3
000000
4
192389124749179824
00000000000000000000000000001
93123717894217819481727182 94
```

输出样例

```
moo~
0
10
2222moo~2000moo~20moo~2moo~moo~moo~11022moo~2moo~2moo~00
1
1moo~2011moo~000moo~12moo~2100101022221221moo~12022111moo~100moo~112
```

HINT

`char` 类型的保存范围是 $[-128, 127]$，尽管此题不会发生，但在面对更大进制转换的问题时（例如 29 进制），由于在处理字符串的过程中，单个数组元素里储存的值可能超过这个值，因此建议储存相关字符串的数组使用 `int` 类型。

为简化在输入上可能对各位造成的困扰，保证输入数据均含有一个空回车符（即和使用键盘输入模拟EOF时一致），同时给出 `int` 型数组的不定组字符串输入代码如下：

```
while(1){ //多组不定组数据输入

    /* 这里应该是每次循环后对变量进行初始化的代码 */

    while((a[i++] = getchar()) != '\n'&& a[i - 1] != EOF); //当getchar()函数没有读到换行符或结束符时，将字符读入数组a
    if(a[i - 1] == EOF) flag = 1;//读到结束符时，标记为此次处理完数据即结束
    if(a[0] < '0' || a[0] > '9'){
        if(flag == 0)
            continue;
        else
            break;
    }//加强代码，以防输入数据里有空回车符或空白符

    /* 这里应该是处理输入字符串的代码 */

    if(flag) //处理标记，flag为1时跳出循环
        break;
}
```

如果执意使用 `char` 类型数组保存，则可以放心使用诸如 `gets` 或 `scanf("%s")` 等函数，同样给出一个输入示例：

```
while(1){//多组不定组数据输入

    /* 这里应该是每次循环后对变量进行初始化的代码 */

    scanf("%s", a);
    if(a[0] == EOF) break;//第0个元素为结束符时跳出循环

   /* 这里应该是处理输入字符串的代码 */

}
```

Author:czy