一起写码吧promax

题目名称	源文件	标准输入文件	标准输出文件	测试点数量	单个测试点分值	时间限制	空间限制
单词背诵	word.cpp	word.in	word.out	10	10	1S	512MB
文件压缩	zip.cpp	zip.in	zip.out	10	10	1S	512MB
会议座位	seat.cpp	seat.in	seat.out	10	10	1S	512MB
最长异或路径	treexor.cpp	treexor.in	treexor.out	10	10	1S	512MB

单词背诵

题目描述

灵梦有 n 个单词想要背，但她想通过一篇文章中的一段来记住这些单词。

文章由 m 个单词构成，她想在文章中找出连续的一段，其中包含最多的她想要背的单词（重复的只算一个）。并且在背诵的单词量尽量多的情况下，还要使选出的文章段落尽量短，这样她就可以用尽量短的时间学习尽可能多的单词了。

输入格式

第 1 行一个数 n，接下来 n 行每行是一个长度不超过 10 的字符串，表示一个要背的单词。

接着是一个数 m，然后是 m 行长度不超过 10 的字符串，每个表示文章中的一个单词。

输出格式

输出文件共 2 行。第 1 行为文章中最多包含的要背的单词数，第 2 行表示在文章中包含最多要背单词的最短的连续段的长度。

样例 #1

样例输入 #1

```
3
hot
dog
milk
5
hot
dog
dog
milk
hot
```

样例输出 #1

```
3
3
```

提示

数据规模与约定

- 对于 30% 的数据，$n \leq 50$，$m \leq 500$；
- 对于 60% 的数据，$n \leq 300$，$m \leq 5000$；
- 对于 100% 的数据，$n \leq 1000$，$m \leq 10^5$。

文件压缩

题目背景

提高文件的压缩率一直是人们追求的目标。近几年有人提出了这样一种算法，它虽然只是单纯地对文件进行重排，本身并不压缩文件，但是经这种算法调整后的文件在大多数情况下都能获得比原来更大的压缩率。

题目描述

该算法具体如下：对一个长度为 n 的字符串 S，首先根据它构造 n 个字符串，其中第 i 个字符串由将 S 的前 $i-1$ 个字符置于末尾得到。然后把这 n 个字符串按照首字符从小到大排序，如果两个字符串的首字符相等，则按照它们**在 S 中的位置**从小到大排序。排序后的字符串的尾字符可以组成一个新的字符串 S'，它的长度也是 n，并且包含了 S 中的每一个字符。最后输出 S' 以及 S 的首字符在 S' 中的位置 p。

举例：S 是 `example`

1. 构造 n 个字符串。

```
example
xamplee
ampleex
mpleexa
pleexam
leexamp
eexampl
```

2. 将字符串排序。

```
ampleex
example
eexampl
leexamp
mpleexa
pleexam
xamplee
```

3. 压缩结果。

$S'=$ `xelpame`，$p=7$

由于英语单词构造的特殊性，某些字母对出现的频率很高，因此在 S' 中相同的字母有很大几率排在一起，从而提高 S' 的压缩率。虽然这种算法利用了英语单词的特性，然而在实践的过程中，人们发现它几乎适用于所有的文件压缩。

请你编一个程序，读入 S' 和 p，输出字符串 S。

保证 S 仅含小写字母（所以输入的 S' 也仅含小写字母）。

输入格式

共三行。

第一行是一个整数 n（$1\leq n\leq 10000$），代表 S' 的长度。

第二行是字符串 S'。

第三行是整数 p。

输出格式

一行，S。

样例 #1

样例输入 #1

```
7
xelpame
7
```

样例输出 #1

```
example
```

会议座位

题目背景

话说校长最近很喜欢召开全校教职工大会，让老师们强行听他装逼

题目描述

现在校长在校园网上公布了一份座位表，n 位老师从左到右依次排成一行。老师们都对这个座位很满意。

然而到了开会时，校长不小心把座位表打乱了，老师们很不满。老师们并不在意自己的位置变了多少，但如果有一对老师 a 和 b，他们原来的座位是 a 在 b 左边，现在变成了 a 在 b 右边，那么这一对老师便会贡献一单位不满值。

校长想知道这些老师的总不满值是多少。

输入格式

第一行一个正整数 n，为 n 位老师。

第二行有 n 个字符串，每个字符串代表老师的名字（大小写敏感）。这一行代表原来的座位表。

第三行有 n 个字符串，代表打乱后的座位表。

输出格式

一行，一个正整数，表示老师们的总不满值。

样例 #1

样例输入 #1

```
3
Stan Kyle Kenny
Kyle Stan Kenny
```

样例输出 #1

```
1
```

样例 #2

样例输入 #2

```
5
A B C D E
B A D E C
```

样例输出 #2

```
3
```

提示

对于 30% 的数据，$1 \leq n \leq 10^3$。

对于 100% 的数据，$1 \leq n \leq 10^5$，每位老师名字长度不超过 5，只有大小写字母并且互不相同。注意名字对大小写敏感。

最长异或路径

题目描述

给定一棵 n 个点的带权树，结点下标从 1 开始到 n。寻找树中找两个结点，求最长的异或路径。

异或路径指的是指两个结点之间唯一路径上的所有边权的异或。

输入格式

第一行一个整数 n，表示点数。

接下来 $n-1$ 行，给出 u, v, w，分别表示树上的 u 点和 v 点有连边，边的权值是 w。

输出格式

一行，一个整数表示答案。

样例 #1

样例输入 #1

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

样例输出 #1

```
7
```

提示

最长异或序列是 $1, 2, 3$，答案是 $7 = 3 \oplus 4$。

数据范围

对30%，$1 \leq n \leq 1000; 0 < u, v \leq n; 0 \leq w < 2^{31}$。

对100%，$1 \leq n \leq 100000; 0 < u, v \leq n; 0 \leq w < 2^{31}$。

为防止ak后的无趣

本老仙特地为各位预留一道附加题

设置1分

觉得想来的可以做一做

题目描述（hack）

这天天气不错，hzhwcmhf神犇给VFleaKing出了一道题：
给你一个长度为N的字符串S，求有多少个不同的长度为L的子串。
子串的定义是S[l]、S[l + 1]、... S[r]这样连续的一段。
两个字符串被认为是不同的当且仅当某个位置上的字符不同。

VFleaKing一看觉得这不是Hash的裸题么！于是果断写了哈希 + 排序。
而hzhwcmhf神犇心里自然知道，这题就是后缀数组的height中 < L的个数 + 1，就是后缀自动机上代表的长度区间包含L的结点个数，就是后缀树深度为L的结点的数量。
但是hzhwcmhf神犇看了看VFleaKing的做法表示非常汗。于是想卡掉他。

VFleaKing使用的是字典序哈希，其代码大致如下：

```
u64 val = 0;
for (int i = 0; i < l; i++)
 val = val * base + s[i] - 'a';
```

u64是无符号int64，范围是[0, 2^64)。VFleaKing让val自然溢出。

base是一个常量，VFleaKing会根据心情决定其值。
VFleaKing还求出来了base ^ l，即base的l次方，这样就能方便地求出所有长度为L的子串的哈希值。
然后VFleaKing给哈希值排序，去重，求出有多少个不同的哈希值，把这个数作为结果。
其算法的C++代码如下：

```
typedef unsigned long long u64;

const int MaxN = 100000;

inline int hash_handle(const char *s, const int &n, const int &l, const int &base)
{
 u64 hash_pow_l = 1;
 for (int i = 1; i <= l; i++)
  hash_pow_l *= base;

 int li_n = 0;
 static u64 li[MaxN];

 u64 val = 0;
 for (int i = 0; i < l; i++)
  val = val * base + s[i] - 'a';
 li[li_n++] = val;
 for (int i = l; i < n; i++)
 {
  val = val * base + s[i] - 'a';
  val -= (s[i - l] - 'a') * hash_pow_l;
  li[li_n++] = val;
 }

 sort(li, li + li_n);
 li_n = unique(li, li + li_n) - li;
 return li_n;
}
```

hzhwcmhf当然知道怎么卡啦！但是他想考考你。

Input

没有输入。

Output

你需要输出一组数据使得VFleaKing的代码WA掉。我们会使用Special Judge检查你的结果的正确性。
输出文件共两行。
第一行两个用空格隔开的数n、l。
第二行是一个长度为n的字符串。只能包含'a'~'z'。
需要保证1 <= n <= 10^5, 1 <= l <= n，
不符合以上格式会WA。
不要有多余字符，很可能导致你WA。