

| 题目名称 | 徐老师的瑞士轮         | 徐老师的羊腿礼盒 | 徐老师的飙车计划 | 徐老师的错误提示  |
|------|-----------------|----------|----------|-----------|
| 文件名  | competition.cpp | gift.cpp | car.cpp  | check.cpp |
| 输入文件 | competition.in  | gift.in  | car.in   | check.in  |
| 输出文件 | competition.out | gift.out | car.out  | check.out |
| 时间限制 | 1000ms          | 1000ms   | 1000ms   | 1000ms    |
| 空间限制 | 512MB           | 512MB    | 512MB    | 512MB     |

# 徐老师的瑞士轮

## 文件读写

- 输入文件competition.in
- 输出文件competition.out

## 限制

- 1000ms
- 128MB

## 题目描述

徐老师准备在学校举办一场瑞士轮比赛，1V1 决斗，比较一下谁的做题能力比较强

但是由于参赛人数过多，徐老师比到后来自己也忘记了具体哪些人和哪些人比过赛，比过哪些项目

参赛选手一共  $n$  人，一共有  $m$  个项目，于是徐老师统计了每个选手已经参加过的项目

对于第  $i$  个选手，用一个长度为  $m$  的 01 串来表示他参加过哪个项目

其中第  $j$  位为 1 则表示这个选手参加过第  $j$  个项目，为 0 则表示这个选手没有参加过第  $j$  个项目

例如一个选手的参赛情况为 0110，说明他参加过第 2 和第 3 个项目

为了比赛能够继续进行，徐老师设定了新的比赛规则——对于第  $j$  个项目，让所有**没有参加过**这个项目的选手和 **参加过** 这个项目的选手分别进行一轮 *PK*

现在徐老师想知道，还需要进行多少轮 *PK* 才能结束比赛？

## 输入格式

输入第一行包含一个整数  $n, m$  表示参赛选手人数和比赛项目数量

接下来  $n$  行, 每行包含一个长度为  $m$  的 01 字符串表示第  $i$  个选手的参赛情况

## 输出格式

输出一个整数表示还需要进行多少轮  $PK$

## 数据范围

对于 30% 的数据满足  $1 \leq n \leq 10, m = 1$

对于 60% 的数据满足  $1 \leq n \leq 1000, 1 \leq m \leq 100$

对于 100% 的数据满足  $1 \leq n \leq 10000, 1 \leq m \leq 100$

## 样例输入1

```
3 3
011
010
100
```

## 样例输出1

```
6
```

## 样例解释1

第 1 个项目有 2 位选手没参加过, 1 位选手参加过, 那么要进行 2 轮  $PK$

第 2 个项目有 1 位选手没参加过, 2 位选手参加过, 那么要进行 2 轮  $PK$

第 3 个项目有 2 位选手没参加过, 1 位选手参加过, 那么要进行 2 轮  $PK$

一共需要进行 6 轮  $PK$

## 样例输入2

```
5 5
10111
01011
11101
11101
01101
```

## 样例输出2

```
20
```

## 样例解释2

每个项目分别要进行 6, 4, 4, 6, 0 轮 *PK*，一共 20 轮

---

# 徐老师的羊腿礼盒

---

## 文件读写

- 输入文件gift.in
- 输出文件gift.out

## 限制

- 1000ms
- 128MB

## 题目描述

逢年过节，送羊腿可是一个非常好的选择，又好吃，又实在

于是徐老师的羊腿店迎来了一波礼盒需求

徐老师为了这一次的活动，特意做了个超大份的《融合羊腿》

这个《融合羊腿》是由各种品种的羊腿拼合而成，羊腿长度为  $len$ ，其中每个单位长度都是一个品种的羊腿

为了方便，我们用小写字母来表示羊腿的品种

例如一个长度为 4 的《融合羊腿》，用  $aabc$  表示，则这个羊腿由 4 段组成，每一段的品种分别为  $a, a, b, c$

现在来了一位顾客，想要购买一段《融合羊腿》当做礼物送给亲戚（也就是从《融合羊腿》中切下连续的一段），但是这位顾客有自己喜欢的一些品种，所以他希望购买 **长度尽可能短** 的《融合羊腿》，并且在这段羊腿中包含他想要的所有品种（不要求顺序）

现在徐老师想知道，这个顾客会买走长度为多少的《融合羊腿》？

## 输入格式

输入第一行包含一个整数  $len$ ，表示《融合羊腿》的长度

输入第二行包含一个长度为  $len$  的字符串  $S1$ ，表示《融合羊腿》的组成情况

输入第三行包含一个整数  $n$ ，表示这位顾客想要的品种数量

输入第四行包含一个正数为  $n$  的字符串  $S2$ ，表示这位顾客想要羊腿品种分别是哪些（保证  $S2$  中只包含小写字母，且不会重复）

## 输出格式

输出一个整数，表示顾客最少买走长度为多少的《融合羊腿》能包含他想要的所有品种

## 数据范围

对于 10% 的数据, 保证  $n = 1, len \leq 26$ ;

对于另外 10% 的数据, 保证  $n = 2, len \leq 26$ , 且  $S2$  中的所有字母在  $S1$  中出现有且仅有一次;

对于另外 10% 的数据, 保证  $n < 26, len \leq 100$ , 且  $S2$  中的所有字母在  $S1$  中出现有且仅有一次;

对于另外 10% 的数据, 保证  $n \leq 26$ , 且  $S1 = S2$ ;

对于另外 10% 的数据, 保证  $n \leq 26, len \leq 100$ ;

对于另外 10% 的数据, 保证  $n \leq 26, len \leq 2000$ ;

对于 100% 的数据, 保证  $1 \leq n \leq 26, n \leq len \leq 10000$ , 且  $S1$  内一定包含  $S2$  内的所有字母。

## 样例输入

```
11
xalaxaxals
3
xls
```

## 样例输出

```
3
```

## 样例解释

比如 `xalas` 是一段包含了顾客想要品种的《融合羊腿》

比如 `lasax` 是一段包含了顾客想要品种的《融合羊腿》

...

其中最短的是 `xls`, 包含了顾客想要品种的《融合羊腿》

# 徐老师的飙车计划

## 文件读写

- 输入文件car.in
- 输出文件car.out

## 限制

- 1000ms
- 128MB

## 题目描述

徐老师最近很喜欢玩《极品飞车》, 体验飙车的快乐, 风驰电掣!

这天他打开了游戏中的一个新地图, 这个地图一共有  $n$  个检查点和  $n - 1$  条道路, 不同的检查点之间可以通过道路直接或间接到达

现在徐老师在这个地图里驾驶着他自己改装的爱车——AMG Mistar1 LaFerrari Aperta Veneno Roadster 疯狂漂移

玩了半天的他回过神来发现这个地图居然需要知道自己通过每个检查点的次数才能通关！

于是他只能向你求助

他希望你能够实现一个动态记录系统，让他在随心所欲漂移的过程中，自动记录每个检查点的通过次数！

徐老师会在每次开始漂移前告诉你他会从  $x$  号检查点一路漂移到达  $y$  号检查点

并且在这之间他会询问你到现在为止某个检查点被通过的次数

## 输入格式

输入第一行包含一个整数  $n$ ，表示检查点数量

接下来  $n - 1$  行，每行包含两个整数  $u, v$  表示  $u, v$  之间存在一条边

接下来一行包含一个整数  $T$  表示徐老师接下来有  $T$  次行为

接下来  $T$  行，每行包含一组数据表示徐老师的一次行为，行为分两种：漂移 和 查询

- 对于 漂移，徐老师会给出三个数据  $1\ x\ y$  表示这次他会从  $x$  号检查点出发到达  $y$  号检查点
- 对于 查询，徐老师会给出两个数据  $2\ x$ ，表示查询到现在为止  $x$  号检查点被他经过的次数

## 输出格式

对于每次 查询，输出结果

## 数据范围

对于 40 的数据， $1 \leq n, T \leq 100$

对于 100 的数据， $1 \leq n, T \leq 1000, 1 \leq x, y \leq n$ 。

## 样例输入

```
4
1 2
2 3
1 4
7
1 3 4
1 1 2
2 1
2 4
1 4 2
2 1
2 4
```

## 样例输出

```
2
1
3
2
```

## 样例解释

第一次漂移徐老师的路径是  $3 \rightarrow 2 \rightarrow 1 \rightarrow 4$

第二次漂移徐老师的路径是  $1 \rightarrow 2$

此时查询 1 号检查点被经过 2 次, 4 号检查点被经过 1 次

第三次漂移徐老师的路径是  $4 \rightarrow 1 \rightarrow 2$

此时查询 1 号检查点被经过 3 次, 4 号检查点被经过 2 次

---

# 徐老师的错误提示

## 文件读写

- 输入文件check.in
- 输出文件check.out

## 限制

- 1000ms
- 512MB

## 题目描述

众所周知, 对于初学 C++ 的同学来说, 最常见的错误就是编译错误

而对于很多初学者的编译错误, 往往是因为语法命令拼写错误

于是这天徐老师准备升级一下 OJ 的系统, 当同学的代码出现编译错误时, 自动判定是否出现了语法拼写错误, 如果存在错误, 则提示错误

为了适配各种语法, 徐老师会给你  $n$  个基础语法表 (每个语法用一个字符串表示)

接着徐老师会给出一个同学代码中的一个语句 (用字符串  $S$  表示)

这里徐老师认为情况只有四种:

1. 这个语句没有错误 (即在徐老师给出的语法表中可以查到这个语法)
2. 这位同学多写了一个字母, 比如把 *for* 打成了 *forr*
3. 这位同学少写了一个字母, 比如把 *for* 打成了 *fr*
4. 这位同学把其中的一个字母写错了, 比如把 *for* 打成了 *far*

现在徐老师需要你的代码对以上四种情况进行判定

## 输入格式

输入第一行, 包含一个整数  $n$  表示基础语法表的语法数量

接下来  $n$  行, 每行包含一个字符串  $S_i$ , 表示一个基础语法

第  $n + 1$  行包含一个字符串  $Q$ , 表示这次要进行判定的同学代码

## 输出格式

输出分三种情况：

1. 若语句没有错误，则输出 *Perfect*
2. 若可以满足 2, 3, 4 三种错误情况，则输出正确的语法语句（如果有多个语法满足条件，则输出语法表中最靠前的那个，即编号最小的那个）
3. 若不满足以上情况，则输出 *Error*

## 数据范围

以下用  $|S|$  表示字符串长度

对于 40% 的数据， $1 \leq |S_i| \leq 50, N \leq 100$ ，其中有 10% 的数据答案为 *Perfect*

对于 100% 的数据， $1 \leq |S_i|, |Q| \leq 1000, N \leq 1000$ ，其中有 10% 的数据答案为 *Error*

特别的，题目保证对于所有字符串，仅包含小写字母，且  $|Q| * \sum S_i \leq 10^8$

## 样例输入1

```
3
for
main
while
fr
```

## 样例输出1

```
for
```

## 样例输入2

```
3
for
main
while
whlle
```

## 样例输出2

```
while
```

## 样例输入3

```
3
for
main
while
mian
```

# 样例输出

Error