Assignment 3 String

本次作业共3道题目。每题设置10个测试案例，其中的5个已经提供用于测试。3道题共计15分。

> **提示**
>
> - 在本次作业中，如果碰到“输入是同一类数据但未知其数目”的情况，可以通过如下方式读取数据：
>
> ```
> string word;
> while (cin >> word) {
>    // do something with word...
> }
> ```
>
> 在每轮循环中，处理当前读入数据。当输入中没有剩余的数据时，循环会终止。
>
> - 在控制台运行程序时，通过如下方式表示输入的结束：
>   - Windows：输入 `Ctrl`+`Z` ，再回车 `Enter`
>   - MacOS 和 Linux: 输入 `Ctrl`+`D`

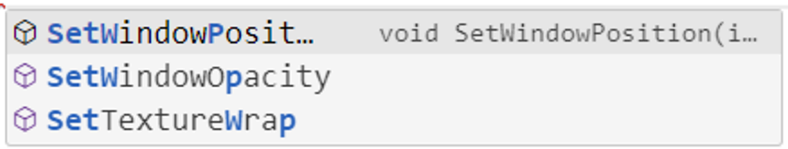
1. 子序列匹配

题目描述

小明在使用编程工具时发现不同IDE的提示机制存在差异：

- **Codeblocks** 采用**前缀匹配**（如输入 `"setwp"` 无法匹配 `"SetWindowPosition"`）
- **VSCode** 支持**子序列匹配**（如输入 `"setwp"` 可匹配 `"SetWindowPosition"`）

```
setwp
```



本题需要同学实现子序列匹配功能。下面定义子序列匹配规则。

子序列匹配条件（不区分大小写）
当满足以下条件时，称目标字符串 `T` 能匹配输入字符串 `S`：

1. `S` 的字符按顺序出现在 `T` 中（允许中间插入其他字符）
2. 每个匹配字符的大小写可忽略（如 `S="swp"` 可匹配 `T="ResetWindowPosition"`）

示例说明：
输入 `S="swp"`，目标 `T="ResetWindowPosition"` 的匹配过程为：

- `S[0] = 's'` → 匹配 `T[2] = 's'`
- `S[1] = 'w'` → 匹配 `T[5] = 'W'`
- `S[2] = 'p'` → 匹配 `T[11] = 'P'`

输入格式

- **第1行**：待匹配的输入字符串 `S`
- **第2行开始**：每行包含用空格分隔的字符串 `T1 T2 T3 ... Tn`（n ≥ 1）

所有字符串的长度 ≥ 1，由字母a-z,A-Z组成。

输出格式

- 按照每行一个字符串的方式，输出能够匹配 `S` 的所有目标字符串 `T`。

输入输出示例

输入:

```
sws
SetWindowTitle SetWindowIcon
SetWindowIcons SetWindowPosition
SetWindowMonitor SetWindowMinSize SetWindowMaxSize
ResetWindowShape SetWindowPosition
```

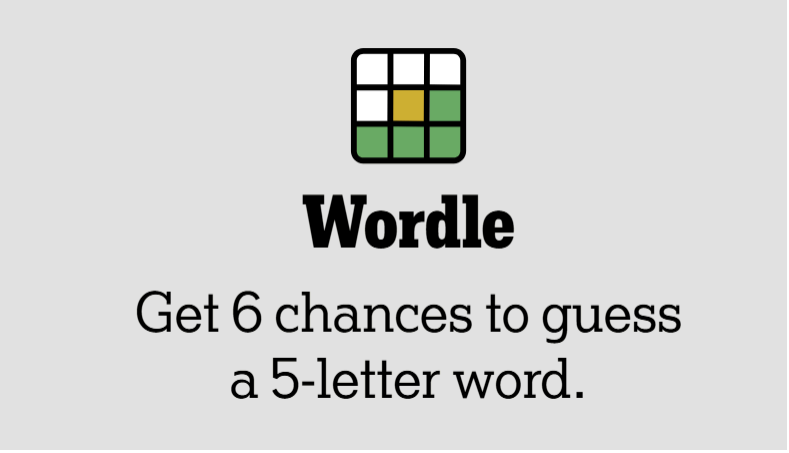
输出:

```
SetWindowIcons
SetWindowPosition
SetWindowMinSize
SetWindowMaxSize
ResetWindowShape
SetWindowPosition
```

2. Wordle

背景

Wordle是一款前几年很火的在线猜词游戏。其核心玩法是玩家需在六次尝试内猜中一个隐藏的五字母英文单词，每次猜测后，系统会通过颜色提示字母的正确性。



玩法

玩家有6次猜测机会，每次猜测必须是一个5个字母的单词。
在每次猜测过后，系统会提示猜测的单词和正确答案的相近程度。

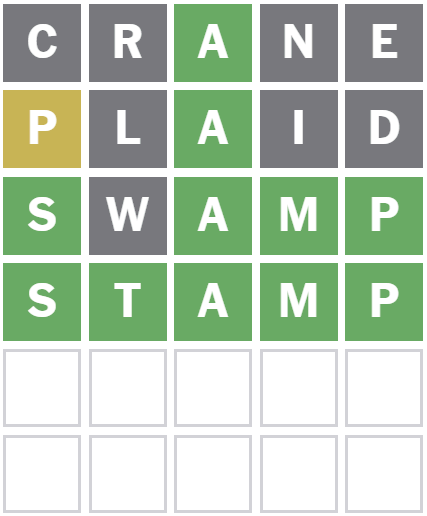
关于"相近程度"的具体描述如下：

针对每一个字母

- 如果该字母在答案中，并且它在正确的位置上，则标为绿色；

- 如果该字母在答案中，但是不在正确的位置上，则标为黄色；
- 如果该字母不在答案中，则标为灰色。

一次示例游戏如下所示：



在答案是"STAMP"的情况下，

1. 第一次猜"CRANE"，字母"A"在"STAMP"中并且在正确的位置，标为绿色，其余字母'C','R','N','E'不在"STAMP"中，标为灰色；
2. 第二次猜"PLAID"，字母"P"在"STAMP"中但是不在正确的位置，标为黄色；
3. 第三和第四次猜测情况以此类推。

题目描述

本题需要同学实现简化版的wordle：读取正确答案并开始一次游戏，根据每轮玩家的猜测反馈结果。
正确答案和玩家的猜测都是长度为5的字符串。

程序**允许**玩家进行至多6轮猜测，玩家的**实际**猜测次数可能小于6。具体流程如下：

1. 对于每一轮猜测，反馈猜测结果。在不区分大小写的前提下（如'a'和'A'表示同一个字母），反馈分为三步：

   1.1 标记完全正确（内容和位置）的字母，这类字母不参与1.2步的“统计剩余字母”；

   1.2 统计剩余字母，标记“存在但位置错误”的字母和“不存在”的字母。

   - 我们说字母`a`“不存在”，是指除去完全正确字母后的答案中不包含字母`a`。
   - 我们说字母`a`“存在但位置错误”，是指除去完全正确字母后的答案中包含字母`a`，但是字母`a`处于猜测单词和答案单词中的不同位置。

   1.3 按照单词从左至右的顺序，输出猜测结果。针对每个字母，

   - 如果该字母完全正确，则输出`g`；
   - 如果该字母存在但位置错误，则输出`y`；
   - 如果该字母不存在，则输出`n`。

2. 若在某一轮猜测中，每个字母都完全正确（即`ggggg`），输出`won`，结束本次游戏。
3. 若经过6轮猜测后，玩家还没有猜中答案，则输出`The answer is` + 全大写字母的答案（如`The answer is STAMP`）。

针对一轮猜测的示例

答案：peach，猜测：CHECK。

1. 标记完全正确的字母："CHECK"中从左往右第二个'C'完全正确，"peach"中的'c'和"CHECK"的第二个'C'不参与第2步标记。
2. 标记“存在但位置错误”的字母和“不存在”的字母（竖着看，检查"CHEK"，此时答案为"peah"）：
    - 'C'不存在
    - 'H'存在但位置错误
    - 'E'存在但位置错误
    - 'K'不存在

因此，CHECK对应的猜测结果是`nyygn`。

输入格式

- **第1行**：长度是5的字符串`s`，作为本次游戏的正确答案。
- **第2行开始**：每行包含一个长度是5的字符串`s'`，表示每轮玩家的猜测。

注意，玩家最多有六次猜测机会，实际猜测次数可能小于6。

字符串`s`和`s'`由英文字母a-z和A-Z组成。

输出格式

按行输出每次猜测的反馈结果。

在达到胜利或失败的条件时，输出提示信息。

输入输出示例

示例1

输入

```
STAMP
CRANE
PLAID
SWAMP
STAMP
```

输出

```
nngnn
yngnn
gnggg
ggggg
won
```

示例2

输入

```
laSSo
crane
avoid
loyal
loupa
lampo
labco
```

输出

```
nnynn
ynynn
gynyn
gynny
ggnng
ggnng
The answer is LASSO
```

3. A=B

背景

A=B是一款编程游戏。
在游戏内，作者实现了一种极简的编程语言。
该语言只有一种指令，即`A=B`，意思是将A替换为B。
一段程序由一系列指令组成，通过**替换**的方式处理字符串。
仅依靠这一种指令，就可以实现许多有意思的功能，比如字符串的去重、排序和比大小。

下面给出一个用`A=B`写的程序，解释它是如何运行的。

示例

假定有下面这段程序：

```
ab=a
a=b
```

- 给定输入串`S="abbba"`，根据指令`ab=a`，程序会从左往右扫描`S`，寻找子串`ab`
- 当第一次出现`ab`时，程序会将`S`中此处`ab`替换为`a`，`S`的值由"**ab**bba"被更新为"**a**bba"，并终止这次扫描
- 每当成功执行一条指令（完成一次替换），程序会**从头开始**重新读取第一行的指令并尝试执行。接下来几次执行，`S`的值的迁移过程为`abbba` → `abba` → `aba` → `aa`
- 当`S=aa`时，执行指令`ab=a`找不到子串`ab`。因此程序会读取第2行的指令`a=b`，并更新`S=aa` → `S=ba`。
- 因为成功执行了`a=b`，程序又会**从头开始**重新读取第一行的指令并执行，执行指令`ab=a`时依然找不到子串`ab`。因此程序会读取第2行的指令`a=b`，并更新`S=ba` → `S=bb`。
- 当没有指令可以更改`S`的值时，程序终止并输出最终的字符串`bb`。

（如果觉得上面这个例子的文字描述太绕，可以看“运行机制”章节对程序运行的精确定义）

直观上理解，上面的这段程序的功能是消除出现在字符`a`后的所有字符`b`，然后将剩下的字符`a`替换为字符`b`。

题目描述

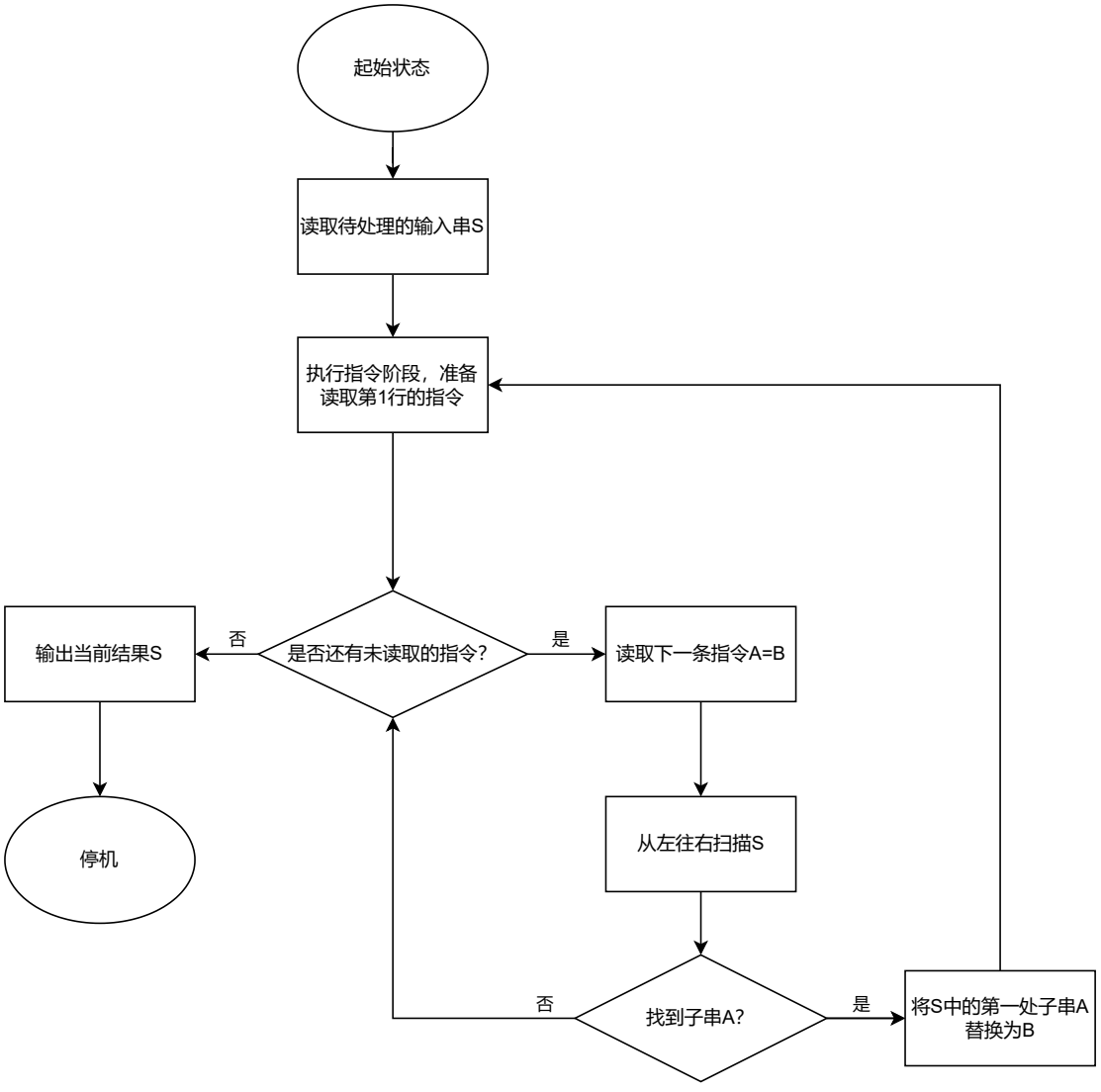
本题需要同学实现这个替换语言`A=B`，读取输入字符串和指令，输出执行指令时字符串内容的变化过程。

运行机制

假设我们将这种编程语言表示的执行模型称为**A=B machine**，它的执行流程如下：

1. 读取输入的待处理字符串`S`
2. 进入执行指令阶段，准备读取第1行的指令
3. 检查是否还有未读取的指令。如果为否，跳转至7
4. 读取下一条指令`A=B`
5. 从左往右扫描`S`，检查是否存在子串`A`。如果为否，跳转至3
6. 将S中的第一处子串`A`替换为`B`，并跳转至2
7. 输出当前结果`S`

文字对应的流程图如下所示。



输入格式

- **第1行**：待处理的输入字符串 `S`
- **第2行开始**：每行包含一条指令，每条指令的格式是`A=B`，其中`A`和`B`是字符串

所有的字符串由字母a-z,A-Z，数字0-9以及其他字符`"[]-.<`组成。`S`的长度 ≥ 1，`A`和`B`可能为空串。

指令数 ≥ 1。

输出格式

根据A=B的运行机制，按行输出程序运行时字符串`S`的变化过程。

注意，我们保证输入的测试用例一定会停机。

输入输出示例

输入：

```
abbba
ab=a
a=b
```

输出：

```
abbba
abba
aba
aa
ba
bb
```

提交格式

在编写完全部程序并测试完毕后，**将3个文件夹中除了main.cpp以外的文件删除之后一起打包压缩**，最后用自己的学号命名，即可提交。
你提交的文件结构应该和以下形式**完全一样**：

```
<your student number>.zip
|- 1_subsequence
|  |- main.cpp
|
|- 2_wordle
|  |- main.cpp
|
|- 3_a_b
|  |- main.cpp
```