C++-字符串-相对开音节构成的结构为辅音+元音
题目描述：
相对开音节构成的结构为辅音+元音（aeiou）+辅音(r 除外)+e，常见的单词有 bike、cake 等。

给定一个字符串，以空格为分隔符，反转每个单词中的字母，若单词中包含如数字等其他非字母时不进行反转。
反转后计算其中含有相对开音节结构的子串个数（连续的子串中部分字符可以重复）。
输入描述：
字符串，以空格分割的多个单词，字符串长度<10000，字母只考虑小写
输出描述：
含有相对开音节结构的子串个数，注：个数<10000
补充说明：
示例 1
输入：
ekam a ekac
输出：
2
说明：
反转后为 make a cake 其中 make、cake 为相对开音节子串，返回 2
示例 2
输入：
!ekam a ekekac
输出：
2
说明：
反转后为!ekam a cakeke 因!ekam 含非英文字符所以未反转，其中 cake、keke 为相对开音节子串，返回 2

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
bool is_yuan(char ch){
    if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') return true;
    return false;
}
bool check(string s){
    if(is_yuan(s[0])) return false;
    if(!is_yuan(s[1])) return false;
    if(is_yuan(s[2]) || s[2] == 'r') return false;
    if(s[3] != 'e') return false;
    return true;
}
int cal(string s){
    string t;
    int n = s.size();
```

```
        bool flag;
        int cnt = 0;
        for(int i = 0; i <= n - 4; i++){
            t.clear();
            flag = true;
            for(int j = 0; j < 4; j++){
                t += s[i + j];
                if(s[i + j] < 'a' || s[i + j] > 'z') flag = false;
            }
            if(flag && check(t)) cnt++;
        }
        return cnt;
}
int main() {
        string s;
        int ans = 0;
        while(cin >> s){
            bool flag = true;
            for(char ch : s){
                if(ch >= 'a' && ch <= 'z') continue;
                flag = false;
            }
            if(flag) reverse(s.begin(), s.end());
            ans += cal(s);
        }
        cout << ans << endl;
}
```