Java-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

```
import java.util.Scanner;

import java.util.*;
// 注意类名必须为 Main, 不要有任何 package xxx 信息
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String[] words = sc.nextLine().split(" ");

        Set<Character> yuan = new HashSet<>();
        yuan.add('a');
        yuan.add('e');
        yuan.add('i');
        yuan.add('o');
        yuan.add('u');
        int cnt = 0;
        for (String word : words) {
```

```
                int i = word.length();
                if (i < 4) continue;
                char[] chars = new char[word.length()];
                for (char c : word.toLowerCase().toCharArray()) {
                        if (c >= 'a' && c <= 'z') {
                                chars[--i] = c;
                        } else {
                                break;
                        }
                }
                if (i > 0) continue;
                while (i + 3 < chars.length) {
                        char first = chars[i];
                        char second = chars[i + 1];
                        char third = chars[i + 2];
                        char fourth = chars[i + 3];
                        if (!yuan.contains(first) && yuan.contains(second) && third != 'r' &&
                                        !yuan.contains(third) && fourth == 'e') {
                                cnt++;
                                i += 2;
                        } else {
                                i++;
                        }
                }
        }
        System.out.println(cnt);
    }
}
```