题目描述

游戏规则：输入一个只包含英文字母的字符串，字符串中的两个字母如果相邻且相同，就可以消除。

在字符串上反复执行消除的动作，直到无法继续消除为止，此时游戏结束。

输出最终得到的字符串长度。

输入描述

输入原始字符串 str ，只能包含大小写英文字母，字母的大小写敏感， str 长度不超过100。

输出描述

输出游戏结束后，最终得到的字符串长度。

备注

输入中包含 非大小写英文字母 时，均为异常输入，直接返回 0。

用例

输入	gg
输出	0
说明	gg 可以直接消除，得到空串，长度为0。

输入	mMbccbc
输出	3
说明	在 mMbccbc 中，可以先消除 cc ； 此时字符串变成 mMbbc ，可以再消除 bb ； 此时字符串变成 mMc，此时没有相邻且相同的字符，无法继续消除。 最终得到的字符串为 mMc ，长度为3。

题目解析

这题比较容易地解法是利用栈结构。

即先创建一个栈，然后遍历输入的字符串，将每一个字符尝试压入栈，但是压入前，需要判断，栈顶元素是否和将要压入的字符相同，若不同，则压入，若相同，则字符不压入，且栈顶元素弹出。

> 由于这题只要遇到两两相邻相同的字符就会消除，比如acccb，消除后就变为了acb，因此遇到>2的奇数个相同相邻字符，是无法消除完的，总是会遗留一个。

注意，遍历的字符如果是非字母，则直接返回0。

Java算法源码

```java
import java.util.LinkedList;
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println(getResult(sc.nextLine()));
  }

  public static int getResult(String s) {
    LinkedList<Character> stack = new LinkedList<>();

    for (int i = 0; i < s.length(); i++) {
      char c = s.charAt(i);

      if (c < 'A' || c > 'z' || (c > 'Z' && c < 'a')) return 0;

      if (stack.size() > 0 && c == stack.getLast()) stack.removeLast();
      else stack.addLast(c);
    }

    return stack.size();
  }
}
```

JS算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

rl.on("line", (line) => {
  console.log(getResult(line));
});

function getResult(s) {
  const stack = [];

  for (let c of s) {
    if (c < "A" || c > "z" || (c > "Z" && c < "a")) return 0;

    if (stack.length > 0 && stack.at(-1) == c) stack.pop();
    else stack.push(c);
  }

  return stack.length;
}
```

Python算法源码

```python
# 输入获取
s = input()


# 算法入口
def getResult():
    stack = []

    for c in s:
        if c < 'A' or c > 'z' or 'Z' < c < 'a':
            return 0

        if len(stack) > 0 and stack[-1] == c:
            stack.pop()
        else:
            stack.append(c)

    return len(stack)


# 算法调用
print(getResult())
```