Given a string `s` of lower and upper case English letters.

A good string is a string which doesn't have **two adjacent characters** `s[i]` and `s[i + 1]` where:

- `0 <= i <= s.length - 2`
- `s[i]` is a lower-case letter and `s[i + 1]` is the same letter but in upper-case or **vice-versa**.

To make the string good, you can choose **two adjacent**characters that make the string bad and remove them. You can keep doing this until the string becomes good.

Return *the string* after making it good. The answer is guaranteed to be unique under the given constraints.

Notice that an empty string is also good.

Example 1:

```
Input: s = "leEeetcode"
Output: "leetcode"
Explanation: In the first step, either you choose i = 1 or i = 2, both will result "leEeetcode"
to be reduced to "leetcode".
```

Example 2:

```
Input: s = "abBAcC"
Output: ""
Explanation: We have many possible scenarios, and all lead to the same answer. For example:
"abBAcC" --> "aAcC" --> "cC" --> ""
"abBAcC" --> "abBA" --> "aA" --> ""
```

Example 3:

```
Input: s = "s"
Output: "s"
```

Constraints:

- `1 <= s.length <= 100`
- `s` contains only lower and upper case English letters.