亲密字符串

给你两个字符串 s 和 goal ，只要我们可以通过交换 s 中的两个字母得到与 goal 相等的结果，就返回 true ；否则返回 false 。

交换字母的定义是：取两个下标 i 和 j （下标从 0 开始）且满足 i != j ，接着交换 s[i] 和 s[j] 处的字符。

- 例如，在 "abcd" 中交换下标 0 和下标 2 的元素可以生成 "cbad" 。

示例 1：

输入： s = "ab", goal = "ba"

输出： true

解释： 你可以交换 s[0] = 'a' 和 s[1] = 'b' 生成 "ba"，此时 s 和 goal 相等。

示例 2：

输入： s = "ab", goal = "ab"

输出： false

解释： 你只能交换 s[0] = 'a' 和 s[1] = 'b' 生成 "ba"，此时 s 和 goal 不相等。

示例 3：

输入： s = "aa", goal = "aa"

输出： true

解释： 你可以交换 s[0] = 'a' 和 s[1] = 'a' 生成 "aa"，此时 s 和 goal 相等。

示例 4：

输入： s = "aaaaaaabc", goal = "aaaaaaacb"

输出： true

```python
class Solution:
    def buddyStrings(self, s: str, goal: str) -> bool:
        #分三种情况讨论
        #情况一：长度不等 直接 false
        if len(s)!=len(goal):
            return False
        #情况二：二者完全相等，但如果字符串不为重复字符，经过一次交换则 false
```

```
        if s==goal and len(set(s))<len(s):
            return True
        #情况三：只有两处不相等
        diffstring=[]
        for i in range(len(s)):
            if s[i]!=goal[i]:
                diffstring.append(i)#记录不相同位置索引
        if len(diffstring)==2:
            if(s[diffstring[0]]==goal[diffstring[1]] and s[diffstring[1]]==
goal[diffstring[0]]):
                return True
            else:
                return False
        return False
```