Problem D. StrangeComparator

Problem Statement

You want to write a revolutionary string comparator that will check whether two strings are equal. You don't want to use the usual comparator because even the best coders make mistakes from time to time. Your new comparator will consider strings s1 and s2 equivalent if they are of equal length, and they differ in at most one position.

Given two String[]s **a** and **b**, return a String[] where the i-th element is equal to "Yes" if **a**[i] and **b**[i] are considered equivalent with your new comparator, or "No" otherwise (all quotes for clarity).

Definition

Class: StrangeComparator
Method: compareString
Parameters: String[], String[]
Returns: String[]
Method signature: String[] compareString(String[] a, String[] b)
(be sure your method is public)

Constraints

- **a** will contain between 1 and 50 elements, inclusive.
- **a** and **b** will contain the same number of elements.
- Each element of **a** will contain only lowercase letters ('a'-'z').
- Each element of **b** will contain only lowercase letters ('a'-'z').
- Each element of **a** will contain between 0 and 50 characters, inclusive.
- Each element of **b** will contain between 0 and 50 characters, inclusive.

Examples

0)

```
{"a", "abc", "abb", "bcd"}

{"ab", "abc", "ade", "bfd"}

Returns: {"No", "Yes", "No", "Yes" }
```

a[0] and **b**[0] are of different length, so they are not equivalent. **a**[2] and **b**[2] are of equal length, but they differ in 2 positions, so they are also not equivalent. The pairs (**a**[1], **b**[1]) and (**a**[3], **b**[3]) are equivalent, because they are of equal length and differ in 0 and 1 positions, respectively.

1)

```
{"", "", "de"}

{"", "a", "bc"}

Returns: {"Yes", "No", "No" }
```

2)

```
{"adcbdeeaeafaklkfajlkfka"}

{"adcbdeearafaklkfajlqfka"}

Returns: {"No" }
```

3)

```
 {"", "abc", "bde", "ahsdjka"}


 {"", "qbp", "fde", "ahsdjka"}


Returns: {"Yes", "No", "Yes", "Yes" }
```