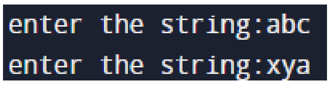
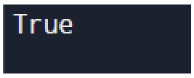
22.Given two strings: s1 and s2 with the same size, check if some permutation of string s1 can break some permutation of string s2 or vice-versa. In other words s2 can break s1 or vice-versa. A string x can break string y (both of size n) if x[i] >= y[i] (in alphabetical order) for all i between 0 and n-1.

PROGRAM:

```
def canPermuteBreaks(s1, s2):
    freq_s1 = {}
    freq_s2 = {}
    for char in s1:
        freq_s1[char] = freq_s1.get(char, 0) + 1
    for char in s2:
        freq_s2[char] = freq_s2.get(char, 0) + 1
    cum_freq_s1 = [0] * 26
    cum_freq_s2 = [0] * 26
    for i in range(26):
        char = chr(ord('a') + i)
        cum_freq_s1[i] = cum_freq_s1[i-1] + freq_s1.get(char, 0)
        cum_freq_s2[i] = cum_freq_s2[i-1] + freq_s2.get(char, 0)
    can_break_s1 = True
    can_break_s2 = True
    for i in range(26):
        if cum_freq_s1[i] < cum_freq_s2[i]:
            can_break_s1 = False
        elif cum_freq_s2[i] < cum_freq_s1[i]:
            can_break_s2 = False
    return can_break_s1 or can_break_s2
s1 = input(“enter the string: “)
s2 = input(“enter the strings: “)
print(canPermuteBreaks(s1, s2))
```

INPUT: 

```
enter the string:abc
enter the string:xya
```

OUTPUT: 

```
True
```

TIMECOMPLEXITY: O(n log n)