796 Rotate String

2018年5月4日 22:55

We are given two strings, A and B.
A *shift on A* consists of taking string A and moving the leftmost character to the rightmost position. For example, if A = 'abcde', then it will be 'bcdea' after one shift on A. Return True if and only if A can become B after some number of shifts on A.
Example 1:
Input: A = 'abcde', B = 'cdeab'
Output: true
Example 2:
Input: A = 'abcde', B = 'abced'
Output: false
Note:

- A and B will have length at most 100.

来自 <https://leetcode.com/problems/rotate-string/description/>
给定两个字符串, A 和 B。
A 的旋转操作就是将 A 最左边的字符移动到最右边。 例如, 若 A = 'abcde'，在移动一次之后结果就是'bcdea' 。如果在若干次旋转操作之后，A 能变成B，那么返回True。
示例 1:
输入: A = 'abcde', B = 'cdeab'
输出: true
示例 2:
输入: A = 'abcde', B = 'abced'
输出: false
注意:

- A 和 B 长度不超过 100。

Solution for Python3:

```python
class Solution:
    def rotateString(self, A, B):
        """
        :type A: str
        :type B: str
        :rtype: bool
        """
        return B in A + A if len(A) == len(B) else False
        # return len(A) == len(B) and B in A + A
```

Solution for C++:

```cpp
class Solution1 {
public:
    bool rotateString(string A, string B) {
        if (A.length() != B.length())
            return false;
        string AA = A + A;
        return  AA.find(B) != string::npos;
    }
```

```
};

class Solution2 {
public:
    bool rotateString(string A, string B) {
        return A.length() == B.length() &&  (A+A).find(B) != string::npos;
    }
};
```