

2. Perform String Shifts You are given a string *s* containing lowercase English letters, and a matrix *shift*, where *shift*[*i*] = [*directioni*, *amounti*]:

- *directioni* can be 0 (for left shift) or 1 (for right shift).
- *amounti* is the amount by which string *s* is to be shifted.
- A left shift by 1 means remove the first character of *s* and append it to the end.
- Similarly, a right shift by 1 means remove the last character of *s* and add it to the beginning.

Return the final string after all operations.

Example 1: Input: *s* = "abc", *shift* = [[0,1],[1,2]] Output: "cab" Explanation: [0,1] means shift to left by 1. "abc" -> "bca" [1,2] means shift to right by 2. "bca" -> "cab"

Example 2: Input: *s* = "abcdefg", *shift* = [[1,1],[1,1],[0,2],[1,3]] Output: "efgabcd" Explanation: [1,1] means shift to right by 1. "abcdefg" -> "gabcdef" [1,1] means shift to right by 1. "gabcdef" -> "fgabcde" [0,2] means shift to left by 2. "fgabcde" -> "abcdefg" [1,3] means shift to right by 3. "abcdefg" -> "efgabcd"

Constraints:

- $1 \leq s.length \leq 100$
- *s* only contains lower case English letters.
- $1 \leq shift.length \leq 100$
- *shift*[*i*].length == 2
- *directioni* is either 0 or 1.
- $0 \leq amounti \leq 100$

PROGRAM:-

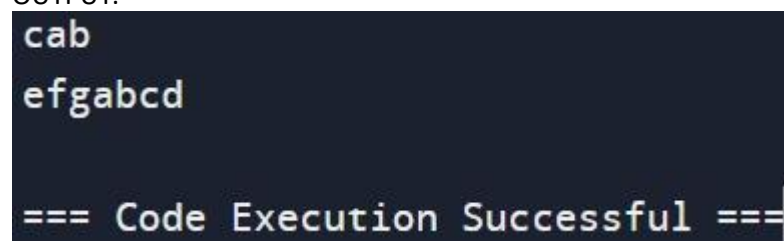
```
def stringShift(s, shift):
    total_shift = 0
    for sh in shift:
        if sh[0] == 0:
            total_shift -= sh[1]
        else:
            total_shift += sh[1]

    total_shift %= len(s)
    return s[-total_shift:] + s[:-total_shift]

# Example 1
s1 = "abc"
shift1 = [[0,1],[1,2]]
print(stringShift(s1, shift1)) # Output: "cab"

# Example 2
s2 = "abcdefg"
shift2 = [[1,1],[1,1],[0,2],[1,3]]
print(stringShift(s2, shift2)) # Output: "efgabcd"
```

OUTPUT:-



```
cab
efgabcd

=== Code Execution Successful ===
```

TIME COMPLEXITY:-O(n)