Perform String Shifts

Solution

You are given a string `s` containing lowercase English letters, and a matrix `shift`, where `shift[i] = [direction, amount]`:

- `direction` can be `0` (for left shift) or `1` (for right shift).
- `amount` 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 <= s.length <= 100`
- `s` only contains lower case English letters.
- `1 <= shift.length <= 100`
- `shift[i].length == 2`
- `0 <= shift[i][0] <= 1`
- `0 <= shift[i][1] <= 100`

Hide Hint #1

Intuitively performing all shift operations is acceptable due to the constraints.

Hide Hint #2

You may notice that left shift cancels the right shift, so count the total left shift times (may be negative if the final result is right shift), and perform it once.

Java

```
/*class Solution {
    public String stringShift(String s, int[][] shift) {
        int move = 0;
        for(int[] sh : shift) {
            move += (sh[0] == 0) ? sh[1] * -1 : sh[1];
        }

        //System.out.println(move);
        if(move % s.length() == 0 || move == 0) return s;

        // if move is positive do right shift
        //char[] array = s.toCharArray();
        String result = "";
        if(move > 0) {
            move = move % s.length();
            move = s.length() - move;
            String first = s.substring(0, move);
            String second = s.substring(move);
            result = second + first;
        }
        // else do left shift
        else {
            move = Math.abs(move);
            move = move % s.length();
            //move = s.length() - move;
            String first = s.substring(0, move);
            String second = s.substring(move);
            result = second + first;
        }

        return result;
    }
}*/

/*class Solution {
    public String stringShift(String s, int[][] shift) {
        int move = 0;
        for(int[] sh : shift) {
            move += (sh[0] == 0) ? sh[1] * -1 : sh[1];
        }

        //System.out.println(move);
        if(move % s.length() == 0 || move == 0) return s;

        // if move is positive do right shift
        char[] array = s.toCharArray();
        String result = "";
        if(move > 0) {
            move = move % s.length();
            for(int i = 0; i < s.length(); i++) {
                array[(i+move)%s.length()] = s.charAt(i);
            }
        }
        // else do left shift
        else {
            move = Math.abs(move);
            move = move % s.length();
            move = s.length() - move;
            move = move % s.length();
            for(int i = 0; i < s.length(); i++) {
                array[(i+move)%s.length()] = s.charAt(i);
            }
        }

        return new String(array);
    }
}*/

class Solution {
    public String stringShift(String s, int[][] shift) {
        int move = 0;
        for(int[] sh : shift) {
            move += (sh[0] == 0) ? sh[1] * -1 : sh[1];
        }

        //System.out.println(move);
        if(move % s.length() == 0 || move == 0) return s;

        // if move is positive do right shift
        char[] array = s.toCharArray();
        String result = "";
        if(move > 0) {
            move = move % s.length();
            for(int i = 0; i < s.length(); i++) {
                array[Math.floorMod((i+move),s.length())] = s.charAt(i);
            }
        }
        // else do left shift
        else {
            move = Math.abs(move);
            move = move % s.length();
            //move = s.length() - move;
            //move = move % s.length();
            for(int i = s.length()-1; i >=0; i--) {
                array[Math.floorMod((i-move),s.length())] = s.charAt(i);
            }
        }

        return new String(array);
    }
}
```

Custom Testcase (Contribute) Run Code Submit

Custom Testcase (Contribute)