∘ ∘cost-climb stairs/)

6114. Move Pieces to Obtain a String

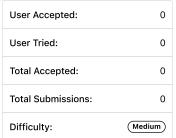
My Submissions (/contest/weekly-contest-301/problems/move-pieces-to-obtain-a-string/submissions/)

Back to Contest (/contest/weekly-contest-301/)

You are given two strings start and target, both of length n. Each string consists only of the characters 'L', 'R', and '_' where:

- The characters 'L' and 'R' represent pieces, where a piece 'L' can move to the **left** only if there is a **blank** space directly to its left, and a piece 'R' can move to the **right** only if there is a **blank** space directly to its right.
- The character '_' represents a blank space that can be occupied by any of the 'L' or 'R' pieces.

Return true if it is possible to obtain the string target by moving the pieces of the string start **any** number of times. Otherwise, return false.



Example 1:

```
Input: start = "_L__R_R_", target = "L____RR"
Output: true
Explanation: We can obtain the string target from start by doing the following moves:
- Move the first piece one step to the left, start becomes equal to "L___R_R_".
- Move the last piece one step to the right, start becomes equal to "L___R_R".
- Move the second piece three steps to the right, start becomes equal to "L____RR".
Since it is possible to get the string target from start, we return true.
```

Example 2:

```
Input: start = "R_L_", target = "__LR"
Output: false
Explanation: The 'R' piece in the string start can move one step to the right to obtain "_RL_".
After that, no pieces can move anymore, so it is impossible to obtain the string target from start.
```

Example 3:

```
Input: start = "_R", target = "R_"
Output: false
Explanation: The piece in the string start can move only to the right, so it is impossible to obtain the string target from start.
```

Constraints:

- n == start.length == target.length
- 1 <= n <= 10^5
- start and target consist of the characters 'L', 'R', and '_'.

```
\boldsymbol{z}
JavaScript
                                                                                                                                 νĎ
 1 \vee \text{const canChange} = (s, t) \Rightarrow \{
         let d = op(s), d2 = op(t);
 3
         if (d.length != d2.length) return false;
 4
         for (let i = 0; i < d.length; i++) {
 5
              let [cs, idx] = d[i], [ct, idx2] = d2[i];
 6
              if (cs != ct) return false;
 7 ▼
              if (cs == 'L') {
 8
                  if (idx < idx2) return false;
 9 •
             } else {
10
                  if (idx > idx2) return false;
11
             }
12
13
         return true;
14
    };
15
16 \cdot const op = (s) \Rightarrow \{
17
         let n = s.length, d = [];
18 •
         for (let i = 0; i < n; i++) {
             if (s[i] == 'L') {
19
```

United States (/region)

```
Move Pieces to Obtain a String - LeetCode Contest
 20
                      d.push(['L', i]);
                } else if (s[i] == 'R') {
    d.push(['R', i]);
 21 •
 22
 23
 24
           }
 25
           return d;
 26
      };
\ \square Custom Testcase
                         Use Example Testcases
                                                                                                                                           Run
                                                                                                                                                       △ Submit
Submission Result: Accepted (/submissions/detail/743160330/) 2
                                                                                      More Details > (/submissions/detail/743160330/)
Share your acceptance!
Copyright © 2022 LeetCode
Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)
```