838. Push Dominoes

Solved

Medium | Topics | Companies

There are `n` dominoes in a line, and we place each domino vertically upright. In the beginning, we simultaneously push some of the dominoes either to the left or to the right.

After each second, each domino that is falling to the left pushes the adjacent domino on the left. Similarly, the dominoes falling to the right push their adjacent dominoes standing on the right.

When a vertical domino has dominoes falling on it from both sides, it stays still due to the balance of the forces.

For the purposes of this question, we will consider that a falling domino expends no additional force to a falling or already fallen domino.

You are given a string `dominoes` representing the initial state where:

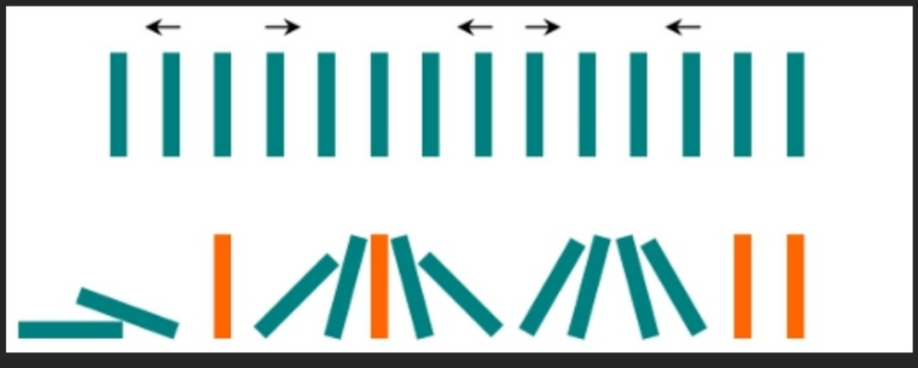
- `dominoes[i] = 'L'`, if the i^{th} domino has been pushed to the left,
- `dominoes[i] = 'R'`, if the i^{th} domino has been pushed to the right, and
- `dominoes[i] = '.'`, if the i^{th} domino has not been pushed.

Return *a string representing the final state*.

Example 1:

> **Input:** dominoes = "RR.L"
> **Output:** "RR.L"
> **Explanation:** The first domino expends no additional force on the second domino.

Example 2:



> **Input:** dominoes = ".L.R...LR..L.."
> **Output:** "LL.RR.LLRRLL.."

Constraints:

- `n == dominoes.length`
- `1 <= n <= 10^5`
- `dominoes[i]` is either `'L'`, `'R'`, or `'.'`.

Accepted 118.2K | Submissions 207.2K | Acceptance Rate 57.1%

.L.R...LR..L...

observation:

case 1 : L L
case 2 : R - ... R
case 3 : L _ ___ R
case 4 : R _ _ _ L
case 5 : L
case 6 : R .....
case 7 : ..... R
case 8 : L .....

} we can handle these cases in another way. insert 'L' at the front and 'R' at the end. Note: These insertions won't make difference or does not affect the situation of dominoes.

new string: "L.L.R...LR..L...R"

now only 4 cases

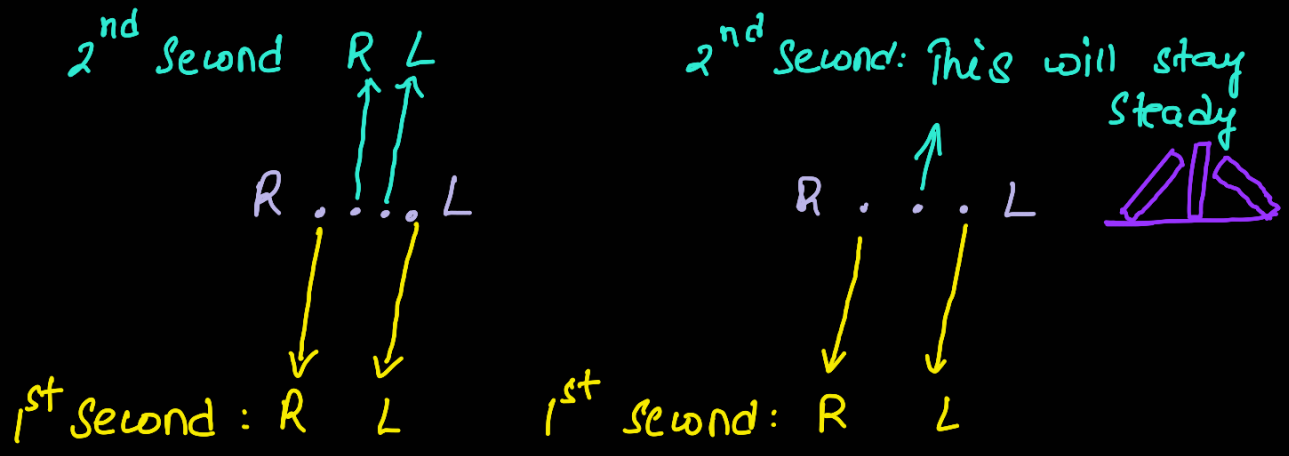
case 1 : L L
case 2 : R _____ R
case 3 : L ____ R
case 4 : R ____ L

→ for case 1 and 2
every dominoes in b/w L & L become L and every dominoe b/w R & R become R.

→ for case 3

No change will be there b/w L & R.

→ for case 4




```
class Solution {
public:
    void equalCase(string &dom,char c,int l,int r){
        int i=l+1;

        while(i<r){
            dom[i]=c;
            i++;
        }

    }

    void differentCase(string &dom,int l,int r){
        int i=l+1,j=r-1;

        while(i<j){
            dom[i]='R';
            dom[j]='L';
            i++,j--;
        }

    }

    string pushDominoes(string dom) {
        dom.insert(dom.begin()+0,'L');
        dom.insert(dom.begin()+dom.length(),'R');

        int i=0,j=1;

        while(j<dom.length()){
            if(dom[j]=='.') j++;
            else{

                if(dom[i] == dom[j]) equalCase(dom,dom[i],i,j);
                else if(dom[i]=='R' && dom[j]=='L') differentCase(dom,i,j);

                j++;
                i=j-1;
            }
        }

        dom.erase(0,1);
        dom.erase(dom.length()-1,1);
        return dom;

    }
};
```

case 1 & 2

case 4

R L

i j

inserting xtra dominoes → O(n)

→ O(n)

These two will run for x no. of times where x is no. of cases in given string.

So Σ no. of characters in x = n

removing xtra dominoes → O(n)

L ..... R L

after handling this we need this R to handle this

$T(n) : O(n) + O(n+n) + O(n)$

$S(n) : O(1)$