2055. Plates Between Candles

My Submissions (/contest/biweekly-contest-64/problems/plates-between-candles/submissions/) Back to Contest (/contest/biweekly-contest-64/)

User Accepted:	1112
User Tried:	1798
Total Accepted:	1143
Total Submissions:	3024
Difficulty:	Medium

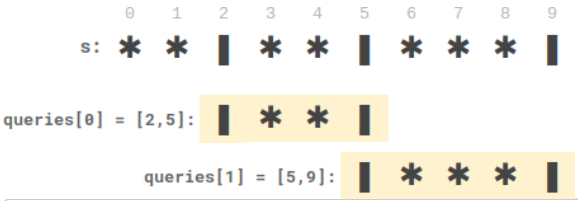
There is a long table with a line of plates and candles arranged on top of it. You are given a **0-indexed** string `s` consisting of characters `'*'` and `'|'` only, where a `'*'` represents a **plate** and a `'|'` represents a **candle**.

You are also given a **0-indexed** 2D integer array `queries` where `queries[i] = [left_i, right_i]` denotes the **substring** `s[left_i...right_i]` (**inclusive**). For each query, you need to find the **number** of plates **between candles** that are **in the substring**. A plate is considered **between candles** if there is at least one candle to its left **and** at least one candle to its right **in the substring**.

- For example, `s = "||**||**|*"`, and a query `[3, 8]` denotes the substring `"*||**|"`. The number of plates between candles in this substring is `2`, as each of the two plates has at least one candle **in the substring** to its left **and** right.

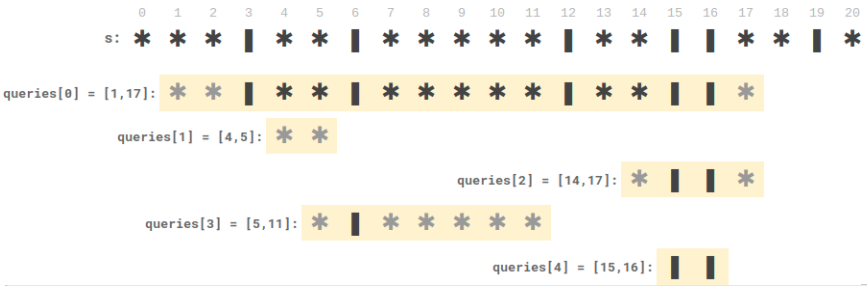
Return *an integer array* `answer` *where* `answer[i]` *is the answer to the* `i^th` *query*.

Example 1:




```
Input: s = "**|**|***|", queries = [[2,5],[5,9]]
Output: [2,3]
Explanation:
- queries[0] has two plates between candles.
- queries[1] has three plates between candles.
```

Example 2:




```
Input: s = "***|**|*****|**||**|*", queries = [[1,17],[4,5],[14,17],[5,11],[15,16]]
Output: [9,0,0,0,0]
Explanation:
- queries[0] has nine plates between candles.
- The other queries have zero plates between candles.
```

Constraints:

- `3 <= s.length <= 10^5`
- `s` consists of `'*'` and `'|'` characters.
- `1 <= queries.length <= 10^5`
- `queries[i].length == 2`
- `0 <= left_i <= right_i < s.length`

Discuss (https://leetcode.com/problems/plates-between-candles/discuss)

JavaScript

```javascript
function Bisect() {
    return { insort_right, insort_left, bisect_left, bisect_right }
    function insort_right(a, x, lo = 0, hi = null) {
        lo = bisect_right(a, x, lo, hi);
        a.splice(lo, 0, x);
    }
    function bisect_right(a, x, lo = 0, hi = null) { // > upper_bound
        if (lo < 0) throw new Error('lo must be non-negative');
        if (hi == null) hi = a.length;
        while (lo < hi) {
            let mid = parseInt((lo + hi) / 2);
            x < a[mid] ? hi = mid : lo = mid + 1;
        }
        return lo;
    }
    function insort_left(a, x, lo = 0, hi = null) {
        lo = bisect_left(a, x, lo, hi);
        a.splice(lo, 0, x);
    }
    function bisect_left(a, x, lo = 0, hi = null) { // >= lower_bound
        if (lo < 0) throw new Error('lo must be non-negative');
        if (hi == null) hi = a.length;
        while (lo < hi) {
            let mid = parseInt((lo + hi) / 2);
            a[mid] < x ? lo = mid + 1 : hi = mid;
        }
        return lo;
    }
}

const cutMaxConsecutive = (a_or_s) => { let d = [], start = 0, n = a_or_s.length; for (let i = 0; i + 1 < n; i++)
{ if (a_or_s[i + 1] != a_or_s[i]) { d.push(a_or_s.slice(start, i + 1)); start = i + 1; } }
d.push(a_or_s.slice(start)); return d; };

const platesBetweenCandles = (ss, queries) => {
    let aa = cutMaxConsecutive(ss);
    let L = [], R = [];
    let preL = 0;
    for (const s of aa) {
        if (s[0] == '*') {
            let l = preL,
                r = l + s.length - 1;
            if (l - 1 >= 0 && r + 1 < ss.length) {
                L.push(l - 1);
                R.push(r + 1);
            }
        }
        preL += s.length;
    }
    let res = [];
    let bi = new Bisect();
    let control = 0;
    for (const [curL, curR] of queries) {
        let cnt = 0;
        let li = bi.bisect_left(L, curL);
        for (let i = li; i < L.length; i++) {
            let l = L[i], r = R[i];
            let len = r - l + 1 - 2;
            if (l > curR) break;
            if (curL <= l && curR >= r) {
                cnt += len;
            }
        }
        res.push(cnt);
    }
    return res;
};
```

Use Example Testcases

Custom Testcase Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/579561536/) More Details (/submissions/detail/579561536/)

Share your acceptance!

Copyright © 2021 LeetCode

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

United States (/region)