Description | Editorial | Solutions (7.4K) | Submissions

387. First Unique Character in a String

Easy 8.1K 261

Companies

Given a string `s`, *find the first non-repeating character in it and return its index*. If it does not exist, return `-1`.

Example 1:

```
Input: s = "leetcode"
Output: 0
```

Example 2:

```
Input: s = "loveleetcode"
Output: 2
```

Example 3:

```
Input: s = "aabb"
Output: -1
```

Constraints:

- `1 <= s.length <= 10^5`
- `s` consists of only lowercase English letters.

Accepted 1.4M | Submissions 2.4M | Acceptance Rate 60.0%

Java | Auto

```java
class Solution {
    public int firstUniqChar(String s) {
        int freq[] = new int [26];
        Queue<Character> q = new LinkedList<>();

        for(int i = 0 ; i < s.length(); i++){
            char ch = s.charAt(i);
            q.add(ch);
            freq[ch-'a']++;

            while(!q.isEmpty() && freq[q.peek()-'a'] >1){
                q.remove();
            }

        }
        return q.isEmpty()?-1 : s.indexOf(q.peek());
```

Ln 18, Col 2

Testcase | Result

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

s =
"leetcode"

Output

0

Expected

Console | Run | Submit

Description | Editorial | Solutions (2.1K) | Submissions

459. Repeated Substring Pattern

Easy 5.7K 459

Companies

Given a string `s`, check if it can be constructed by taking a substring of it and appending multiple copies of the substring together.

Example 1:

```
Input: s = "abab"
Output: true
Explanation: It is the substring "ab" twice.
```

Example 2:

```
Input: s = "aba"
Output: false
```

Example 3:

```
Input: s = "abcabcabcabc"
Output: true
Explanation: It is the substring "abc" four times or the substring
"abcabc" twice.
```

Constraints:

- `1 <= s.length <= 10^4`
- `s` consists of lowercase English letters.

Java | Auto

```java
public class Solution {
    public boolean repeatedSubstringPattern(String s) {
        int n = s.length();
        for (int i = 1; i <= n / 2; i++) {
            if (n % i == 0) {
                String substring = s.substring(0, i);
                StringBuilder repeated = new StringBuilder();
                for (int j = 0; j < n / i; j++) {
                    repeated.append(substring);
                }
                if (repeated.toString().equals(s)) return true;
            }
        }
        return false;
    }
}
```

Ln 16, Col 2

Testcase | Result

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

s =
"abab"

Output

true

Expected

Console Run Submit