05 March 2025 11:14

Question 1 ...

# 3. Longest Substring Without Repeating Characters

Medium | Topics | Companies | Hint

Given a string `s`, find the length of the **longest substring** without duplicate characters.

**Example 1:**

```
Input: s = "abcabcbb"
Output: 3
Explanation: The answer is "abc", with the length of 3.
```

**Example 2:**

```
Input: s = "bbbbb"
Output: 1
Explanation: The answer is "b", with the length of 1.
```

Brute force ..

```java
public int lengthOfLongestSubstring(String s) {
    int[] arr=new int[256];
    int maxl=0;

    for(int i=0;i<s.length();i++){
        for(int j=i;j<s.length();j++){
            if(arr[s.charAt(j)]!=0){
                arr[s.charAt(j)]=0;
                break;
            }
            arr[s.charAt(j)]++;
            maxl=Math.max(maxl , j-i+1);
        }
        Arrays.fill(arr,0);
    }
    return maxl;
}
```

Input

s =
" "

correct by ... int[26]

Input

s =
"dvdf"

Output

3

correct by .. Array again define to zero ...

Arrays.fill (arr,0)

Better approach :

```java
public int lengthOfLongestSubstring(String s) {
    int[] arr=new int[256];
    int maxl=0;
    int l=0 , r=0;

    while(r<s.length()){

        while(arr[s.charAt(r)]!=0){
            arr[s.charAt(l)]--;
            l++;
        }

        if(arr[s.charAt(r)]==0){
            arr[s.charAt(r)]++;
            maxl=Math.max(maxl , r-l+1);
        }
        r++;
    }

    return maxl;
}
```

Optimal approach

```java
public int lengthOfLongestSubstring(String s) {
    int[] arr = new int[256]; // Frequency array to track character occurrences
    int maxl = 0; // Stores the maximum length of substring
    int l = 0, r = 0; // Left and Right pointers

    while (r < s.length()) {
        // If the character at r is already present in the current window
        if (arr[s.charAt(r)] != 0) {
            // Move l to the next position after the last occurrence of s.charAt(r)
            l = Math.max(l, arr[s.charAt(r)]);
        }

        // Store the next index (right + 1) of the character to avoid shrinking
        arr[s.charAt(r)] = r + 1;

        // Calculate the max window length
        maxl = Math.max(maxl, r - l + 1);

        r++; // Move right pointer
    }

    return maxl;
}
```

Instant of storing the ...
... is reppeat .... Store where it is store.....

Input

s =
"pwwkew"

Output

3

.... Solve by stroing the index ...instant of one two in array ....

Problem with

**l = Math.max(l, arr[s.charAt(r)]);**

s =
"abba"

Output

2