# **Leetcode 3 – Longest Substring Without Repeating Characters**

## Problem Understanding

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

* The characters must be **contiguous** (i.e., part of a substring).
* The substring must contain **only unique characters**.

### Example

Input: s = "abcabcbb"

Output: 3

Explanation: The answer is "abc", with length 3.

## Optimized Java Solution (Sliding Window + Freq Array)

class Solution {

public int lengthOfLongestSubstring(String s) {

int[] freq = new int[128]; // ASCII character frequency

int left = 0, right = 0;

int maxLen = 0;

while (right < s.length()) {

char c = s.charAt(right);

freq[c]++;

// Shrink window if duplicate found

while (freq[c] > 1) {

freq[s.charAt(left)]--;

left++;

}

maxLen = Math.max(maxLen, right - left + 1);

right++;

}

return maxLen;

}

}

## Dry Run Using Table

### Input:

s = "abcabcbb"

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Left | Right | Window | Action | maxLen |
| 1 | 0 | 0 | "a" | Add 'a' | 1 |
| 2 | 0 | 1 | "ab" | Add 'b' | 2 |
| 3 | 0 | 2 | "abc" | Add 'c' | 3 |
| 4 | 0 | 3 | "abca" | Duplicate 'a' → move left to 1 | 3 |
| 5 | 1 | 3 | "bca" | Continue | 3 |
| 6 | 1 | 4 | "bcab" | Duplicate 'b' → move left to 2 | 3 |
| 7 | 2 | 4 | "cab" | Continue | 3 |
| 8 | 2 | 5 | "cabc" | Duplicate 'c' → move left to 3 | 3 |
| 9 | 3 | 5 | "abc" | Continue | 3 |
| ... | ... | ... | ... | Repeat for final "b" | 3 |

✅ Final Answer: 3

## Time / Space Complexity

|  |  |
| --- | --- |
| Metric | Value |
| Time | O(n) |
| Space | O(128) = O(1) |

* Because we use a fixed-size frequency array for ASCII characters.

## Alternate Approaches

|  |  |  |  |
| --- | --- | --- | --- |
| Approach | Time | Space | Notes |
| ✅ Sliding Window | O(n) | O(1) | Best & optimal solution |
| Brute Force | O(n²) | O(n) | Check all substrings, slow |
| HashSet with While | O(n) | O(n) | Also valid with set sliding win |