Problem 159: Longest Substring with At Most Two Distinct Characters

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

Given a string

s

, return

the length of the longest

substring

that contains at most

two distinct characters

.

Example 1:

Input:

s = "eceba"

Output:

3

Explanation:

The substring is "ece" which its length is 3.

Example 2:

Input:

s = "ccaabbb"

Output:

5

Explanation:

The substring is "aabbb" which its length is 5.

Constraints:

1 <= s.length <= 10

5

s

consists of English letters.

Code Snippets

C++:

```
class Solution {
public:
int lengthOfLongestSubstringTwoDistinct(string s) {

}
};
```

Java:

```
class Solution {
public int lengthOfLongestSubstringTwoDistinct(String s) {

}
}
```

Python3:

```
class Solution:
def lengthOfLongestSubstringTwoDistinct(self, s: str) -> int:
```

Python:

```
class Solution(object):
def lengthOfLongestSubstringTwoDistinct(self, s):
"""
:type s: str
:rtype: int
"""
```

JavaScript:

```
/**
* @param {string} s
* @return {number}
*/
var lengthOfLongestSubstringTwoDistinct = function(s) {

};
```

TypeScript:

```
function lengthOfLongestSubstringTwoDistinct(s: string): number {

};
```

C#:

```
public class Solution {
public int LengthOfLongestSubstringTwoDistinct(string s) {
```

```
}
}
```

C:

```
int lengthOfLongestSubstringTwoDistinct(char* s) {

}
```

Go:

```
func lengthOfLongestSubstringTwoDistinct(s string) int {

}
```

Kotlin:

```
class Solution {
fun lengthOfLongestSubstringTwoDistinct(s: String): Int {

}
}
```

Swift:

```
class Solution {
func lengthOfLongestSubstringTwoDistinct(_ s: String) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn length_of_longest_substring_two_distinct(s: String) -> i32 {

}
}
```

Ruby:

```
# @param {String} s
# @return {Integer}
def length_of_longest_substring_two_distinct(s)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @return Integer
*/
function lengthOfLongestSubstringTwoDistinct($s) {

}
}
```

Dart:

```
class Solution {
int lengthOfLongestSubstringTwoDistinct(String s) {

}
}
```

Scala:

```
object Solution {
def lengthOfLongestSubstringTwoDistinct(s: String): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec length_of_longest_substring_two_distinct(s :: String.t) :: integer
def length_of_longest_substring_two_distinct(s) do

end
end
```

Erlang:

```
-spec length_of_longest_substring_two_distinct(S :: unicode:unicode_binary())
-> integer().
length_of_longest_substring_two_distinct(S) ->
.
```

Racket:

```
(define/contract (length-of-longest-substring-two-distinct s)
(-> string? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Longest Substring with At Most Two Distinct Characters
* Difficulty: Medium
* Tags: array, string, tree, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public:
int lengthOfLongestSubstringTwoDistinct(string s) {

}
};
```

Java Solution:

```
/**
* Problem: Longest Substring with At Most Two Distinct Characters
* Difficulty: Medium
* Tags: array, string, tree, hash
*
```

```
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public int lengthOfLongestSubstringTwoDistinct(String s) {

}
}
```

Python3 Solution:

```
"""
Problem: Longest Substring with At Most Two Distinct Characters
Difficulty: Medium
Tags: array, string, tree, hash

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(h) for recursion stack where h is height
"""

class Solution:
def lengthOfLongestSubstringTwoDistinct(self, s: str) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def lengthOfLongestSubstringTwoDistinct(self, s):
"""
:type s: str
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Longest Substring with At Most Two Distinct Characters
```

```
* Difficulty: Medium
* Tags: array, string, tree, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

/**
* @param {string} s
* @return {number}
*/
var lengthOfLongestSubstringTwoDistinct = function(s) {

};
```

TypeScript Solution:

```
/**
* Problem: Longest Substring with At Most Two Distinct Characters
* Difficulty: Medium
* Tags: array, string, tree, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

function lengthOfLongestSubstringTwoDistinct(s: string): number {

};
```

C# Solution:

```
/*
* Problem: Longest Substring with At Most Two Distinct Characters
* Difficulty: Medium
* Tags: array, string, tree, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
```

```
* Space Complexity: O(h) for recursion stack where h is height
*/

public class Solution {
public int LengthOfLongestSubstringTwoDistinct(string s) {

}
}
```

C Solution:

```
/*
* Problem: Longest Substring with At Most Two Distinct Characters
* Difficulty: Medium
* Tags: array, string, tree, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

int lengthOfLongestSubstringTwoDistinct(char* s) {

}
```

Go Solution:

```
// Problem: Longest Substring with At Most Two Distinct Characters
// Difficulty: Medium
// Tags: array, string, tree, hash
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

func lengthOfLongestSubstringTwoDistinct(s string) int {

}
```

Kotlin Solution:

```
class Solution {
fun lengthOfLongestSubstringTwoDistinct(s: String): Int {

}
}
```

Swift Solution:

```
class Solution {
func lengthOfLongestSubstringTwoDistinct(_ s: String) -> Int {

}
}
```

Rust Solution:

```
// Problem: Longest Substring with At Most Two Distinct Characters
// Difficulty: Medium
// Tags: array, string, tree, hash
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

impl Solution {
pub fn length_of_longest_substring_two_distinct(s: String) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String} s
# @return {Integer}
def length_of_longest_substring_two_distinct(s)

end
```

PHP Solution:

```
class Solution {
```

```
/**
* @param String $s
* @return Integer
*/
function lengthOfLongestSubstringTwoDistinct($s) {

}
}
```

Dart Solution:

```
class Solution {
int lengthOfLongestSubstringTwoDistinct(String s) {

}
}
```

Scala Solution:

```
object Solution {
def lengthOfLongestSubstringTwoDistinct(s: String): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec length_of_longest_substring_two_distinct(s :: String.t) :: integer
def length_of_longest_substring_two_distinct(s) do

end
end
```

Erlang Solution:

```
-spec length_of_longest_substring_two_distinct(S :: unicode:unicode_binary())
-> integer().
length_of_longest_substring_two_distinct(S) ->
.
```

Racket Solution:

```
(define/contract (length-of-longest-substring-two-distinct s)
(-> string? exact-integer?)
)
```