

# Problem 2083: Substrings That Begin and End With the Same Letter

## Problem Information

Difficulty: **Medium**

Acceptance Rate: 0.00%

Paid Only: No

## Problem Description

You are given a

0-indexed

string

s

consisting of only lowercase English letters. Return

the number of

substrings

in

s

that begin and end with the

same

character.

A

substring

is a contiguous non-empty sequence of characters within a string.

Example 1:

Input:

s = "abcba"

Output:

7

Explanation:

The substrings of length 1 that start and end with the same letter are: "a", "b", "c", "b", and "a".  
The substring of length 3 that starts and ends with the same letter is: "bcb". The substring of length 5 that starts and ends with the same letter is: "abcba".

Example 2:

Input:

s = "abacad"

Output:

9

Explanation:

The substrings of length 1 that start and end with the same letter are: "a", "b", "a", "c", "a", and "d". The substrings of length 3 that start and end with the same letter are: "aba" and "aca".  
The substring of length 5 that starts and ends with the same letter is: "abaca".

Example 3:

Input:

s = "a"

Output:

1

Explanation:

The substring of length 1 that starts and ends with the same letter is: "a".

Constraints:

1 <= s.length <= 10

5

s

consists only of lowercase English letters.

## Code Snippets

**C++:**

```
class Solution {  
public:  
    long long numberOfSubstrings(string s) {  
  
    }  
};
```

**Java:**

```
class Solution {  
    public long numberOfSubstrings(String s) {  
  
    }  
}
```

### Python3:

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

### Python:

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

### JavaScript:

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

};
```

### TypeScript:

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

};
```

### C#:

```
public class Solution {
    public long NumberOfSubstrings(string s) {

    }
}
```

### C:

```
long long numberOfSubstrings(char* s) {

}
```

**Go:**

```
func numberOfSubstrings(s string) int64 {  
  
}
```

**Kotlin:**

```
class Solution {  
    fun numberOfSubstrings(s: String): Long {  
  
    }  
}
```

**Swift:**

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

**Rust:**

```
impl Solution {  
    pub fn number_of_substrings(s: String) -> i64 {  
  
    }  
}
```

**Ruby:**

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

**PHP:**

```
class Solution {  
  
    /**
```

```

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

}

}

```

#### Dart:

```

class Solution {
  int numberOfSubstrings(String s) {

  }

}

```

#### Scala:

```

object Solution {
  def numberOfSubstrings(s: String): Long = {

  }

}

```

#### Elixir:

```

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

  end

end

```

#### Erlang:

```

-spec number_of_substrings(S :: unicode:unicode_binary()) -> integer().
number_of_substrings(S) ->
.

```

#### Racket:

```
(define/contract (number-of-substrings s)
  (-> string? exact-integer?)
)
```

## Solutions

### C++ Solution:

```
/*
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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:
    long long numberOfSubstrings(string s) {

    }
};
```

### Java Solution:

```
/**
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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 long numberOfSubstrings(String s) {

    }
}
```

```
}
```

### Python3 Solution:

```
"""
Problem: Substrings That Begin and End With the Same Letter
Difficulty: Medium
Tags: array, string, tree, math, 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 numberOfSubstrings(self, s: str) -> int:
        # TODO: Implement optimized solution
        pass
```

### Python Solution:

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

### JavaScript Solution:

```
/**
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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 numberOfSubstrings = function(s) {

};

```

### TypeScript Solution:

```

/**
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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 numberOfSubstrings(s: string): number {

};

```

### C# Solution:

```

/*
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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 long NumberOfSubstrings(string s) {

    }
}

```

### C Solution:

```
/*
 * Problem: Substrings That Begin and End With the Same Letter
 * Difficulty: Medium
 * Tags: array, string, tree, math, 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
 */

long long numberOfSubstrings(char* s) {

}
```

### Go Solution:

```
// Problem: Substrings That Begin and End With the Same Letter
// Difficulty: Medium
// Tags: array, string, tree, math, 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 numberOfSubstrings(s string) int64 {

}
```

### Kotlin Solution:

```
class Solution {
    fun numberOfSubstrings(s: String): Long {

    }
}
```

### Swift Solution:

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

```
}  
}
```

### Rust Solution:

```
// Problem: Substrings That Begin and End With the Same Letter  
// Difficulty: Medium  
// Tags: array, string, tree, math, 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 number_of_substrings(s: String) -> i64 {  
  
    }  
}
```

### Ruby Solution:

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

### PHP Solution:

```
class Solution {  
  
    /**  
     * @param String $s  
     * @return Integer  
     */  
    function numberOfSubstrings($s) {  
  
    }  
}
```

### Dart Solution:

```
class Solution {  
  int numberOfSubstrings(String s) {  
  
  }  
}
```

### Scala Solution:

```
object Solution {  
  def numberOfSubstrings(s: String): Long = {  
  
  }  
}
```

### Elixir Solution:

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

### Erlang Solution:

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

### Racket Solution:

```
(define/contract (number-of-substrings s)  
  (-> string? exact-integer?)  
)
```