Problem 2955: Number of Same-End Substrings

Problem Information

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

Problem Description

You are given a

0-indexed

string

s

, and a 2D array of integers

queries

, where

queries[i] = [l

i

, r

i

]

indicates a substring of

s

starting from the index

l

i

and ending at the index

r

i

(both

inclusive

), i.e.

s[l

i

..r

i

]

.

Return

an array

ans

where

ans[i]

is the number of

same-end

substrings of

queries[i]

.

A

0-indexed

string

t

of length

n

is called

same-end

if it has the same character at both of its ends, i.e.,

t[0] == t[n - 1]

.

A

substring

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

Example 1:

Input:

s = "abcaab", queries = [[0,0],[1,4],[2,5],[0,5]]

Output:

[1,5,5,10]

Explanation:

Here is the same-end substrings of each query: 1

st

query: s[0..0] is "a" which has 1 same-end substring: "

a

". 2

nd

query: s[1..4] is "bcaa" which has 5 same-end substrings: "

b

caa", "b

c

aa", "bc

a

a", "bca

a

", "bc

aa

". 3

rd

query: s[2..5] is "caab" which has 5 same-end substrings: "

c

aab", "c

a

ab", "ca

a

b", "caa

b

", "c

aa

b". 4

th

query: s[0..5] is "abcaab" which has 10 same-end substrings: "

a

bcaab", "a

b

caab", "ab

c

aab", "abc

a

ab", "abca

a

b", "abcaa

b

", "abc

aa

b", "

abca

ab", "

abcaa

b", "a

bcaab

".

Example 2:

Input:

s = "abcd", queries = [[0,3]]

Output:

[4]

Explanation:

The only query is s[0..3] which is "abcd". It has 4 same-end substrings: "

a

bcd", "a

b

cd", "ab

c

d", "abc

d

".

Constraints:

2 <= s.length <= 3 * 10

4

s

consists only of lowercase English letters.

1 <= queries.length <= 3 * 10

4

queries[i] = [l

i

, r

i

]

0 <= l

i

<= r

i

< s.length

Code Snippets

C++:

```
class Solution {
public:
vector<int> sameEndSubstringCount(string s, vector<vector<int>>& queries) {

}
};
```

Java:

```
class Solution {
public int[] sameEndSubstringCount(String s, int[][] queries) {
}
}
```

Python3:

```
class Solution:
def sameEndSubstringCount(self, s: str, queries: List[List[int]]) ->
List[int]:
```

Python:

```
class Solution(object):
def sameEndSubstringCount(self, s, queries):
"""
:type s: str
:type queries: List[List[int]]
:rtype: List[int]
"""
```

JavaScript:

```
/**
* @param {string} s
* @param {number[][]} queries
* @return {number[]}
*/
var sameEndSubstringCount = function(s, queries) {

};
```

TypeScript:

```
function sameEndSubstringCount(s: string, queries: number[][]): number[] {

};
```

C#:

```
public class Solution {
public int[] SameEndSubstringCount(string s, int[][] queries) {

}
}
```

C:

```
/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* sameEndSubstringCount(char* s, int** queries, int queriesSize, int*
queriesColSize, int* returnSize) {

}
```

Go:

```
func sameEndSubstringCount(s string, queries [][]int) []int {

}
```

Kotlin:

```
class Solution {
fun sameEndSubstringCount(s: String, queries: Array<IntArray>): IntArray {

}
}
```

Swift:

```
class Solution {
func sameEndSubstringCount(_ s: String, _ queries: [[Int]]) -> [Int] {

}
}
```

Rust:

```
impl Solution {
pub fn same_end_substring_count(s: String, queries: Vec<Vec<i32>>) ->
Vec<i32> {

}

}
```

Ruby:

```
# @param {String} s
# @param {Integer[][]} queries
# @return {Integer[]}
def same_end_substring_count(s, queries)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @param Integer[][] $queries
* @return Integer[]
*/
function sameEndSubstringCount($s, $queries) {

}
}
```

Dart:

```
class Solution {
List<int> sameEndSubstringCount(String s, List<List<int>> queries) {

}
}
```

Scala:

```
object Solution {
def sameEndSubstringCount(s: String, queries: Array[Array[Int]]): Array[Int]
= {

}
}
```

Elixir:

```
defmodule Solution do
@spec same_end_substring_count(s :: String.t, queries :: [[integer]]) ::
[integer]
def same_end_substring_count(s, queries) do

end
end
```

Erlang:

```
-spec same_end_substring_count(S :: unicode:unicode_binary(), Queries ::
[[integer()]]) -> [integer()].
same_end_substring_count(S, Queries) ->
.
```

Racket:

```
(define/contract (same-end-substring-count s queries)
(-> string? (listof (listof exact-integer?)) (listof exact-integer?))
)
```

Solutions

C++ Solution:

```
/*
* Problem: Number of Same-End Substrings
* 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:
vector<int> sameEndSubstringCount(string s, vector<vector<int>>& queries) {

}
};
```

Java Solution:

```
/**
* Problem: Number of Same-End Substrings
* 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[] sameEndSubstringCount(String s, int[][] queries) {

}
}
```

Python3 Solution:

```
"""
Problem: Number of Same-End Substrings
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 sameEndSubstringCount(self, s: str, queries: List[List[int]]) ->
List[int]:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def sameEndSubstringCount(self, s, queries):
"""
:type s: str
:type queries: List[List[int]]
:rtype: List[int]
"""
```

JavaScript Solution:

```
/**
* Problem: Number of Same-End Substrings
* 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
* @param {number[][]} queries
* @return {number[]}
*/
var sameEndSubstringCount = function(s, queries) {

};
```

TypeScript Solution:

```
/**
* Problem: Number of Same-End Substrings
* 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 sameEndSubstringCount(s: string, queries: number[][]): number[] {
};
```

C# Solution:

```
/*
* Problem: Number of Same-End Substrings
* 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[] SameEndSubstringCount(string s, int[][] queries) {

}
}
```

C Solution:

```
/*
* Problem: Number of Same-End Substrings
* 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
*/

/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* sameEndSubstringCount(char* s, int** queries, int queriesSize, int*
queriesColSize, int* returnSize) {

}
```

Go Solution:

```go
// Problem: Number of Same-End Substrings
// 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 sameEndSubstringCount(s string, queries [][]int) []int {

}
```

Kotlin Solution:

```kotlin
class Solution {
fun sameEndSubstringCount(s: String, queries: Array<IntArray>): IntArray {

}
}
```

Swift Solution:

```swift
class Solution {
func sameEndSubstringCount(_ s: String, _ queries: [[Int]]) -> [Int] {

}
}
```

Rust Solution:

```rust
// Problem: Number of Same-End Substrings
// 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 same_end_substring_count(s: String, queries: Vec<Vec<i32>>) ->
Vec<i32> {

}
}
```

Ruby Solution:

```
# @param {String} s
# @param {Integer[][]} queries
# @return {Integer[]}
def same_end_substring_count(s, queries)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $s
* @param Integer[][] $queries
* @return Integer[]
*/
function sameEndSubstringCount($s, $queries) {

}
}
```

Dart Solution:

```
class Solution {
List<int> sameEndSubstringCount(String s, List<List<int>> queries) {

}
}
```

Scala Solution:

```
object Solution {
def sameEndSubstringCount(s: String, queries: Array[Array[Int]]): Array[Int]
= {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec same_end_substring_count(s :: String.t, queries :: [[integer]]) ::
[integer]
def same_end_substring_count(s, queries) do

end
end
```

Erlang Solution:

```
-spec same_end_substring_count(S :: unicode:unicode_binary(), Queries ::
[[integer()]]) -> [integer()].
same_end_substring_count(S, Queries) ->
.
```

Racket Solution:

```
(define/contract (same-end-substring-count s queries)
(-> string? (listof (listof exact-integer?)) (listof exact-integer?))
)
```