Problem 2062: Count Vowel Substrings of a String

Problem Information

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

Problem Description

A

substring

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

A

vowel substring

is a substring that

only

consists of vowels (

'a'

,

'e'

,

'i'

,

'o'

, and

'u'

) and has

all five

vowels present in it.

Given a string

word

, return

the number of

vowel substrings

in

word

.

Example 1:

Input:

word = "aeiouu"

Output:

2

Explanation:

The vowel substrings of word are as follows (underlined): - "

aeiou

u" - "

aeiouu

"

Example 2:

Input:

word = "unicornarihan"

Output:

0

Explanation:

Not all 5 vowels are present, so there are no vowel substrings.

Example 3:

Input:

word = "cuaieuouac"

Output:

7

Explanation:

The vowel substrings of word are as follows (underlined): - "c

uaieuo

uac" - "c

uaieuou

ac" - "c

uaieuoua

c" - "cu

aieuo

uac" - "cu

aieuou

ac" - "cu

aieuoua

c" - "cua

ieuoua

c"

Constraints:

1 <= word.length <= 100

word

consists of lowercase English letters only.

Code Snippets

C++:

```
class Solution {
public:
int countVowelSubstrings(string word) {

}
};
```

Java:

```
class Solution {
public int countVowelSubstrings(String word) {

}
}
```

Python3:

```
class Solution:
def countVowelSubstrings(self, word: str) -> int:
```

Python:

```
class Solution(object):
def countVowelSubstrings(self, word):
"""
:type word: str
:rtype: int
"""
```

JavaScript:

```
/**
* @param {string} word
* @return {number}
*/
var countVowelSubstrings = function(word) {

};
```

TypeScript:

```
function countVowelSubstrings(word: string): number {

};
```

C#:

```
public class Solution {
public int CountVowelSubstrings(string word) {

}
}
```

C:

```
int countVowelSubstrings(char* word) {

}
```

Go:

```
func countVowelSubstrings(word string) int {

}
```

Kotlin:

```
class Solution {
fun countVowelSubstrings(word: String): Int {

}
}
```

Swift:

```
class Solution {
func countVowelSubstrings(_ word: String) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn count_vowel_substrings(word: String) -> i32 {

}
}
```

Ruby:

```
# @param {String} word
# @return {Integer}
def count_vowel_substrings(word)

end
```

PHP:

```
class Solution {

/**
* @param String $word
* @return Integer
*/
function countVowelSubstrings($word) {

}
}
```

Dart:

```
class Solution {
int countVowelSubstrings(String word) {

}
}
```

Scala:

```
object Solution {
def countVowelSubstrings(word: String): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec count_vowel_substrings(word :: String.t) :: integer
def count_vowel_substrings(word) do

end
end
```

Erlang:

```
-spec count_vowel_substrings(Word :: unicode:unicode_binary()) -> integer().
count_vowel_substrings(Word) ->
.
```

Racket:

```
(define/contract (count-vowel-substrings word)
(-> string? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public:
int countVowelSubstrings(string word) {
}
};
```

Java Solution:

```
/**
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public int countVowelSubstrings(String word) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Vowel Substrings of a String
Difficulty: Easy
Tags: string, tree, hash

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(h) for recursion stack where h is height
"""

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

Python Solution:

```
class Solution(object):
def countVowelSubstrings(self, word):
"""
:type word: str
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

/**
* @param {string} word
* @return {number}
*/
var countVowelSubstrings = function(word) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

function countVowelSubstrings(word: string): number {

};
```

C# Solution:

```
/*
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* 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 CountVowelSubstrings(string word) {

}
}
```

C Solution:

```
/*
* Problem: Count Vowel Substrings of a String
* Difficulty: Easy
* Tags: string, tree, hash
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

int countVowelSubstrings(char* word) {

}
```

Go Solution:

```
// Problem: Count Vowel Substrings of a String
// Difficulty: Easy
// Tags: string, tree, hash
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

func countVowelSubstrings(word string) int {

}
```

Kotlin Solution:

```
class Solution {
fun countVowelSubstrings(word: String): Int {

}
}
```

Swift Solution:

```
class Solution {
func countVowelSubstrings(_ word: String) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Vowel Substrings of a String
// Difficulty: Easy
// Tags: string, tree, hash
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

impl Solution {
pub fn count_vowel_substrings(word: String) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String} word
# @return {Integer}
def count_vowel_substrings(word)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $word
* @return Integer
*/
function countVowelSubstrings($word) {

}
}
```

Dart Solution:

```
class Solution {
int countVowelSubstrings(String word) {

}
}
```

Scala Solution:

```
object Solution {
def countVowelSubstrings(word: String): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec count_vowel_substrings(word :: String.t) :: integer
def count_vowel_substrings(word) do

end
```

```
end
```

Erlang Solution:

```
-spec count_vowel_substrings(Word :: unicode:unicode_binary()) -> integer().
count_vowel_substrings(Word) ->
.
```

Racket Solution:

```
(define/contract (count-vowel-substrings word)
(-> string? exact-integer?)
)
```