Problem 2278: Percentage of Letter in String

Problem Information

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

Problem Description

Given a string

s

and a character

letter

, return

the

percentage

of characters in

s

that equal

letter

rounded down

to the nearest whole percent.

Example 1:

Input:

s = "foobar", letter = "o"

Output:

33

Explanation:

The percentage of characters in s that equal the letter 'o' is 2 / 6 * 100% = 33% when rounded down, so we return 33.

Example 2:

Input:

s = "jjjj", letter = "k"

Output:

0

Explanation:

The percentage of characters in s that equal the letter 'k' is 0%, so we return 0.

Constraints:

1 <= s.length <= 100

s

consists of lowercase English letters.

letter

is a lowercase English letter.

Code Snippets

C++:

```
class Solution {
public:
int percentageLetter(string s, char letter) {

}
};
```

Java:

```
class Solution {
public int percentageLetter(String s, char letter) {

}
}
```

Python3:

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

Python:

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

JavaScript:

```
/**
* @param {string} s
* @param {character} letter
* @return {number}
*/
var percentageLetter = function(s, letter) {

};
```

TypeScript:

```
function percentageLetter(s: string, letter: string): number {

};
```

C#:

```
public class Solution {
public int PercentageLetter(string s, char letter) {

}
}
```

C:

```
int percentageLetter(char* s, char letter) {

}
```

Go:

```
func percentageLetter(s string, letter byte) int {

}
```

Kotlin:

```
class Solution {
fun percentageLetter(s: String, letter: Char): Int {

}
}
```

Swift:

```
class Solution {
func percentageLetter(_ s: String, _ letter: Character) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn percentage_letter(s: String, letter: char) -> i32 {

}
}
```

Ruby:

```
# @param {String} s
# @param {Character} letter
# @return {Integer}
def percentage_letter(s, letter)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @param String $letter
* @return Integer
*/
function percentageLetter($s, $letter) {

}
}
```

Dart:

```
class Solution {
int percentageLetter(String s, String letter) {

}

}
```

Scala:

```
object Solution {
def percentageLetter(s: String, letter: Char): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec percentage_letter(s :: String.t, letter :: char) :: integer
def percentage_letter(s, letter) do

end
end
```

Erlang:

```
-spec percentage_letter(S :: unicode:unicode_binary(), Letter :: char()) ->
integer().
percentage_letter(S, Letter) ->
.
```

Racket:

```
(define/contract (percentage-letter s letter)
(-> string? char? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
```

```
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
int percentageLetter(string s, char letter) {

}
};
```

Java Solution:

```
/**
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int percentageLetter(String s, char letter) {

}
}
```

Python3 Solution:

```
"""
Problem: Percentage of Letter in String
Difficulty: Easy
Tags: string

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
```

```
"""

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

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {string} s
* @param {character} letter
* @return {number}
*/
var percentageLetter = function(s, letter) {

};
```

TypeScript Solution:

```
/**
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function percentageLetter(s: string, letter: string): number {

};
```

C# Solution:

```
/*
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int PercentageLetter(string s, char letter) {

}
}
```

C Solution:

```
/*
* Problem: Percentage of Letter in String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
```

```
*/

int percentageLetter(char* s, char letter) {

}
```

Go Solution:

```
// Problem: Percentage of Letter in String
// Difficulty: Easy
// Tags: string
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func percentageLetter(s string, letter byte) int {

}
```

Kotlin Solution:

```
class Solution {
fun percentageLetter(s: String, letter: Char): Int {

}
}
```

Swift Solution:

```
class Solution {
func percentageLetter(_ s: String, _ letter: Character) -> Int {

}
}
```

Rust Solution:

```
// Problem: Percentage of Letter in String
// Difficulty: Easy
// Tags: string
```

```
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn percentage_letter(s: String, letter: char) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String} s
# @param {Character} letter
# @return {Integer}
def percentage_letter(s, letter)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $s
* @param String $letter
* @return Integer
*/
function percentageLetter($s, $letter) {

}
}
```

Dart Solution:

```
class Solution {
int percentageLetter(String s, String letter) {

}
}
```

Scala Solution:

```
object Solution {
def percentageLetter(s: String, letter: Char): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec percentage_letter(s :: String.t, letter :: char) :: integer
def percentage_letter(s, letter) do

end
end
```

Erlang Solution:

```
-spec percentage_letter(S :: unicode:unicode_binary(), Letter :: char()) ->
integer().
percentage_letter(S, Letter) ->
.
```

Racket Solution:

```
(define/contract (percentage-letter s letter)
(-> string? char? exact-integer?)
)
```