Problem 3582: Generate Tag for Video Caption

Problem Information

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

Problem Description

You are given a string

caption

representing the caption for a video.

The following actions must be performed

in order

to generate a

valid tag

for the video:

Combine all words

in the string into a single

camelCase string

prefixed with

'#'

. A

camelCase string

is one where the first letter of all words

except

the first one is capitalized. All characters after the first character in

each

word must be lowercase.

Remove

all characters that are not an English letter,

except

the first

'#'

.

Truncate

the result to a maximum of 100 characters.

Return the

tag

after performing the actions on

caption

.

Example 1:

Input:

caption = "Leetcode daily streak achieved"

Output:

"#leetcodeDailyStreakAchieved"

Explanation:

The first letter for all words except

"leetcode"

should be capitalized.

Example 2:

Input:

caption = "can I Go There"

Output:

"#canIGoThere"

Explanation:

The first letter for all words except

"can"

should be capitalized.

Example 3:

Input:

caption = "hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh"

Output:

"#hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh"

Explanation:

Since the first word has length 101, we need to truncate the last two letters from the word.

Constraints:

1 <= caption.length <= 150

caption

consists only of English letters and

' '

.

Code Snippets

C++:

```
class Solution {
public:
string generateTag(string caption) {

}
};
```

Java:

```
class Solution {
public String generateTag(String caption) {

}
}
```

Python3:

```
class Solution:
def generateTag(self, caption: str) -> str:
```

Python:

```
class Solution(object):
def generateTag(self, caption):
"""
:type caption: str
:rtype: str
"""
```

JavaScript:

```
/**
* @param {string} caption
* @return {string}
*/
var generateTag = function(caption) {

};
```

TypeScript:

```
function generateTag(caption: string): string {

};
```

C#:

```
public class Solution {
public string GenerateTag(string caption) {

}
}
```

C:

```
char* generateTag(char* caption) {

}
```

Go:

```
func generateTag(caption string) string {

}
```

Kotlin:

```
class Solution {
fun generateTag(caption: String): String {

}
}
```

Swift:

```
class Solution {
func generateTag(_ caption: String) -> String {

}
}
```

Rust:

```
impl Solution {
pub fn generate_tag(caption: String) -> String {

}
}
```

Ruby:

```
# @param {String} caption
# @return {String}
def generate_tag(caption)

end
```

PHP:

```
class Solution {

/**
* @param String $caption
* @return String
*/
function generateTag($caption) {

}
}
```

Dart:

```
class Solution {
String generateTag(String caption) {

}
}
```

Scala:

```
object Solution {
def generateTag(caption: String): String = {

}
}
```

Elixir:

```
defmodule Solution do
@spec generate_tag(caption :: String.t) :: String.t
def generate_tag(caption) do

end
end
```

Erlang:

```
-spec generate_tag(Caption :: unicode:unicode_binary()) ->
unicode:unicode_binary().
generate_tag(Caption) ->
```

```
.
```

Racket:

```
(define/contract (generate-tag caption)
(-> string? string?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Generate Tag for Video Caption
* 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:
string generateTag(string caption) {

}
};
```

Java Solution:

```
/**
* Problem: Generate Tag for Video Caption
* 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 String generateTag(String caption) {

}
}
```

Python3 Solution:

```
"""
Problem: Generate Tag for Video Caption
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 generateTag(self, caption: str) -> str:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def generateTag(self, caption):
"""
:type caption: str
:rtype: str
"""
```

JavaScript Solution:

```
/**
* Problem: Generate Tag for Video Caption
* 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} caption
* @return {string}
*/
var generateTag = function(caption) {

};
```

TypeScript Solution:

```
/**
* Problem: Generate Tag for Video Caption
* 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 generateTag(caption: string): string {

};
```

C# Solution:

```
/*
* Problem: Generate Tag for Video Caption
* 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 string GenerateTag(string caption) {

}
}
```

C Solution:

```
/*
* Problem: Generate Tag for Video Caption
* 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
*/

char* generateTag(char* caption) {

}
```

Go Solution:

```
// Problem: Generate Tag for Video Caption
// 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 generateTag(caption string) string {

}
```

Kotlin Solution:

```
class Solution {
fun generateTag(caption: String): String {

}
}
```

Swift Solution:

```
class Solution {
func generateTag(_ caption: String) -> String {

}
}
```

Rust Solution:

```
// Problem: Generate Tag for Video Caption
// 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 generate_tag(caption: String) -> String {

}
}
```

Ruby Solution:

```
# @param {String} caption
# @return {String}
def generate_tag(caption)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $caption
* @return String
*/
function generateTag($caption) {

}
}
```

Dart Solution:

```
class Solution {
String generateTag(String caption) {

}
}
```

Scala Solution:

```
object Solution {
def generateTag(caption: String): String = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec generate_tag(caption :: String.t) :: String.t
def generate_tag(caption) do

end
end
```

Erlang Solution:

```
-spec generate_tag(Caption :: unicode:unicode_binary()) ->
unicode:unicode_binary().
generate_tag(Caption) ->
.
```

Racket Solution:

```
(define/contract (generate-tag caption)
(-> string? string?)
)
```