

Problem 3441: Minimum Cost Good Caption

Problem Information

Difficulty: **Hard**

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

You are given a string

caption

of length

n

. A

good

caption is a string where

every

character appears in groups of

at least 3

consecutive occurrences.

For example:

"aaabbb"

and

"aaaaccc"

are

good

captions.

"aabbb"

and

"ccccd"

are

not

good captions.

You can perform the following operation

any

number of times:

Choose an index

i

(where

$0 \leq i < n$

) and change the character at that index to either:

The character immediately

before

it in the alphabet (if

`caption[i] != 'a'`

).

The character immediately

after

it in the alphabet (if

`caption[i] != 'z'`

).

Your task is to convert the given

caption

into a

good

caption using the

minimum

number of operations, and return it. If there are

multiple

possible good captions, return the

lexicographically smallest

one among them. If it is

impossible

to create a good caption, return an empty string

""

.

Example 1:

Input:

caption = "cdcd"

Output:

"cccc"

Explanation:

It can be shown that the given caption cannot be transformed into a good caption with fewer than 2 operations. The possible good captions that can be created using exactly 2 operations are:

"dddd"

: Change

caption[0]

and

caption[2]

to their next character

'd'

.

"cccc"

: Change

caption[1]

and

caption[3]

to their previous character

'c'

.

Since

"cccc"

is lexicographically smaller than

"dddd"

, return

"cccc"

.

Example 2:

Input:

caption = "aca"

Output:

"aaa"

Explanation:

It can be proven that the given caption requires at least 2 operations to be transformed into a good caption. The only good caption that can be obtained with exactly 2 operations is as follows:

Operation 1: Change

caption[1]

to

'b'

.

caption = "aba"

.

Operation 2: Change

caption[1]

to

'a'

.

caption = "aaa"

.

Thus, return

"aaa"

.

Example 3:

Input:

caption = "bc"

Output:

""

Explanation:

It can be shown that the given caption cannot be converted to a good caption by using any number of operations.

Constraints:

$1 \leq \text{caption.length} \leq 5 * 10$

4

caption

consists only of lowercase English letters.

Code Snippets

C++:

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

    }
}
```

```
};
```

Java:

```
class Solution {  
    public String minCostGoodCaption(String caption) {  
  
    }  
}
```

Python3:

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

Python:

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

JavaScript:

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

TypeScript:

```
function minCostGoodCaption(caption: string): string {  
  
};
```

C#:

```
public class Solution {  
    public string MinCostGoodCaption(string caption) {  
  
    }  
}
```

C:

```
char* minCostGoodCaption(char* caption) {  
  
}
```

Go:

```
func minCostGoodCaption(caption string) string {  
  
}
```

Kotlin:

```
class Solution {  
    fun minCostGoodCaption(caption: String): String {  
  
    }  
}
```

Swift:

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

Rust:

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

Ruby:

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

end
```

PHP:

```
class Solution {

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

    }

}
```

Dart:

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

  }
}
```

Scala:

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

  }
}
```

Elixir:

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

  end
end
```

Erlang:

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

Racket:

```
(define/contract (min-cost-good-caption caption)
  (-> string? string?)
)
```

Solutions

C++ Solution:

```
/*
 * Problem: Minimum Cost Good Caption
 * Difficulty: Hard
 * Tags: string, graph, dp
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

class Solution {
public:
    string minCostGoodCaption(string caption) {

    }
};
```

Java Solution:

```
/**
 * Problem: Minimum Cost Good Caption
 * Difficulty: Hard
 * Tags: string, graph, dp
 *

```

```

* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

class Solution {
public String minCostGoodCaption(String caption) {

}

}

```

Python3 Solution:

```

"""
Problem: Minimum Cost Good Caption
Difficulty: Hard
Tags: string, graph, dp

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

class Solution:
def minCostGoodCaption(self, caption: str) -> str:
# TODO: Implement optimized solution
pass

```

Python Solution:

```

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

```

JavaScript Solution:

```

/**
* Problem: Minimum Cost Good Caption

```

```

* Difficulty: Hard
* Tags: string, graph, dp
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

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

};

```

TypeScript Solution:

```

/**
* Problem: Minimum Cost Good Caption
* Difficulty: Hard
* Tags: string, graph, dp
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

function minCostGoodCaption(caption: string): string {

};

```

C# Solution:

```

/*
* Problem: Minimum Cost Good Caption
* Difficulty: Hard
* Tags: string, graph, dp
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)

```

```

* Space Complexity: O(n) or O(n * m) for DP table
*/

public class Solution {
    public string MinCostGoodCaption(string caption) {

    }
}

```

C Solution:

```

/*
* Problem: Minimum Cost Good Caption
* Difficulty: Hard
* Tags: string, graph, dp
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

char* minCostGoodCaption(char* caption) {

}

```

Go Solution:

```

// Problem: Minimum Cost Good Caption
// Difficulty: Hard
// Tags: string, graph, dp
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

func minCostGoodCaption(caption string) string {

}

```

Kotlin Solution:

```

class Solution {
    fun minCostGoodCaption(caption: String): String {

    }
}

```

Swift Solution:

```

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

    }
}

```

Rust Solution:

```

// Problem: Minimum Cost Good Caption
// Difficulty: Hard
// Tags: string, graph, dp
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

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

    }
}

```

Ruby Solution:

```

# @param {String} caption
# @return {String}
def min_cost_good_caption(caption)

end

```

PHP Solution:

```

class Solution {

```

```

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

}
}

```

Dart Solution:

```

class Solution {
  String minCostGoodCaption(String caption) {

  }
}

```

Scala Solution:

```

object Solution {
  def minCostGoodCaption(caption: String): String = {

  }
}

```

Elixir Solution:

```

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

  end
end

```

Erlang Solution:

```

-spec min_cost_good_caption(Caption :: unicode:unicode_binary()) ->
  unicode:unicode_binary().
min_cost_good_caption(Caption) ->
.

```

Racket Solution:

```
(define/contract (min-cost-good-caption caption)
  (-> string? string?)
)
```