Problem 800: Similar RGB Color

Problem Information

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

Problem Description

The red-green-blue color

"#AABBCC"

can be written as

"#ABC"

in shorthand.

For example,

"#15c"

is shorthand for the color

"#1155cc"

.

The similarity between the two colors

"#ABCDEF"

and

"#UVWXYZ"

is

-(AB - UV)

2

- (CD - WX)

2

- (EF - YZ)

2

.

Given a string

color

that follows the format

"#ABCDEF"

, return a string represents the color that is most similar to the given color and has a shorthand (i.e., it can be represented as some

"#XYZ"

).

Any answer

which has the same highest similarity as the best answer will be accepted.

Example 1:

Input:

color = "#09f166"

Output:

"#11ee66"

Explanation:

The similarity is -(0x09 - 0x11)

2

-(0xf1 - 0xee)

2

- (0x66 - 0x66)

2

= -64 -9 -0 = -73. This is the highest among any shorthand color.

Example 2:

Input:

color = "#4e3fe1"

Output:

"#5544dd"

Constraints:

color.length == 7

color[0] == '#'

color[i]

is either digit or character in the range

['a', 'f']

for

i > 0

.

Code Snippets

C++:

```
class Solution {
public:
string similarRGB(string color) {

}
};
```

Java:

```
class Solution {
public String similarRGB(String color) {

}
}
```

Python3:

```
class Solution:
def similarRGB(self, color: str) -> str:
```

Python:

```
class Solution(object):
def similarRGB(self, color):
```

```
"""
:type color: str
:rtype: str
"""
```

JavaScript:

```
/**
* @param {string} color
* @return {string}
*/
var similarRGB = function(color) {

};
```

TypeScript:

```
function similarRGB(color: string): string {

};
```

C#:

```
public class Solution {
public string SimilarRGB(string color) {

}
}
```

C:

```
char* similarRGB(char* color) {

}
```

Go:

```
func similarRGB(color string) string {

}
```

Kotlin:

```
class Solution {
fun similarRGB(color: String): String {

}
}
```

Swift:

```
class Solution {
func similarRGB(_ color: String) -> String {

}
}
```

Rust:

```
impl Solution {
pub fn similar_rgb(color: String) -> String {

}
}
```

Ruby:

```
# @param {String} color
# @return {String}
def similar_rgb(color)

end
```

PHP:

```
class Solution {

/**
* @param String $color
* @return String
*/
function similarRGB($color) {

}
}
```

Dart:

```
class Solution {
String similarRGB(String color) {

}
}
```

Scala:

```
object Solution {
def similarRGB(color: String): String = {

}
}
```

Elixir:

```
defmodule Solution do
@spec similar_rgb(color :: String.t) :: String.t
def similar_rgb(color) do

end
end
```

Erlang:

```
-spec similar_rgb(Color :: unicode:unicode_binary()) ->
unicode:unicode_binary().
similar_rgb(Color) ->
.
```

Racket:

```
(define/contract (similar-rgb color)
(-> string? string?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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 similarRGB(string color) {

}
};
```

Java Solution:

```
/**
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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 similarRGB(String color) {

}
}
```

Python3 Solution:

```
"""
Problem: Similar RGB Color
Difficulty: Easy
Tags: string, math
```

```
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 similarRGB(self, color: str) -> str:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def similarRGB(self, color):
"""
:type color: str
:rtype: str
"""
```

JavaScript Solution:

```
/**
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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} color
* @return {string}
*/
var similarRGB = function(color) {

};
```

TypeScript Solution:

```
/**
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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 similarRGB(color: string): string {

};
```

C# Solution:

```
/*
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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 SimilarRGB(string color) {

}
}
```

C Solution:

```
/*
* Problem: Similar RGB Color
* Difficulty: Easy
* Tags: string, math
*
* 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* similarRGB(char* color) {

}
```

Go Solution:

```
// Problem: Similar RGB Color
// Difficulty: Easy
// Tags: string, math
//
// 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 similarRGB(color string) string {

}
```

Kotlin Solution:

```
class Solution {
fun similarRGB(color: String): String {

}
}
```

Swift Solution:

```
class Solution {
func similarRGB(_ color: String) -> String {

}
}
```

Rust Solution:

```
// Problem: Similar RGB Color
// Difficulty: Easy
// Tags: string, math
```

```
//
// 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 similar_rgb(color: String) -> String {

}
}
```

Ruby Solution:

```
# @param {String} color
# @return {String}
def similar_rgb(color)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $color
* @return String
*/
function similarRGB($color) {

}
}
```

Dart Solution:

```
class Solution {
String similarRGB(String color) {

}
}
```

Scala Solution:

```
object Solution {
def similarRGB(color: String): String = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec similar_rgb(color :: String.t) :: String.t
def similar_rgb(color) do

end
end
```

Erlang Solution:

```
-spec similar_rgb(Color :: unicode:unicode_binary()) ->
unicode:unicode_binary().
similar_rgb(Color) ->
.
```

Racket Solution:

```
(define/contract (similar-rgb color)
(-> string? string?)
)
```