Problem 3461: Check If Digits Are Equal in String After Operations I

Problem Information

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

Problem Description

You are given a string

s

consisting of digits. Perform the following operation repeatedly until the string has

exactly

two digits:

For each pair of consecutive digits in

s

, starting from the first digit, calculate a new digit as the sum of the two digits

modulo

10.

Replace

s

with the sequence of newly calculated digits,

maintaining the order

in which they are computed.

Return

true

if the final two digits in

s

are the

same

; otherwise, return

false

.

Example 1:

Input:

s = "3902"

Output:

true

Explanation:

Initially,

s = "3902"

First operation:

(s[0] + s[1]) % 10 = (3 + 9) % 10 = 2

(s[1] + s[2]) % 10 = (9 + 0) % 10 = 9

(s[2] + s[3]) % 10 = (0 + 2) % 10 = 2

s

becomes

"292"

Second operation:

(s[0] + s[1]) % 10 = (2 + 9) % 10 = 1

(s[1] + s[2]) % 10 = (9 + 2) % 10 = 1

s

becomes

"11"

Since the digits in

"11"

are the same, the output is

true

.

Example 2:

Input:

s = "34789"

Output:

false

Explanation:

Initially,

s = "34789"

.

After the first operation,

s = "7157"

.

After the second operation,

s = "862"

.

After the third operation,

s = "48"

.

Since

'4' != '8'

, the output is

false

.

Constraints:

3 <= s.length <= 100

s

consists of only digits.

Code Snippets

C++:

```
class Solution {
public:
bool hasSameDigits(string s) {

}
};
```

Java:

```
class Solution {
public boolean hasSameDigits(String s) {

}
}
```

Python3:

```
class Solution:
def hasSameDigits(self, s: str) -> bool:
```

Python:

```
class Solution(object):
def hasSameDigits(self, s):
```

```
"""
:type s: str
:rtype: bool
"""
```

JavaScript:

```
/**
* @param {string} s
* @return {boolean}
*/
var hasSameDigits = function(s) {

};
```

TypeScript:

```
function hasSameDigits(s: string): boolean {

};
```

C#:

```
public class Solution {
public bool HasSameDigits(string s) {

}
}
```

C:

```
bool hasSameDigits(char* s) {

}
```

Go:

```
func hasSameDigits(s string) bool {

}
```

Kotlin:

```
class Solution {
fun hasSameDigits(s: String): Boolean {

}
}
```

Swift:

```
class Solution {
func hasSameDigits(_ s: String) -> Bool {

}
}
```

Rust:

```
impl Solution {
pub fn has_same_digits(s: String) -> bool {

}
}
```

Ruby:

```
# @param {String} s
# @return {Boolean}
def has_same_digits(s)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @return Boolean
*/
function hasSameDigits($s) {

}
}
```

Dart:

```
class Solution {
bool hasSameDigits(String s) {

}
}
```

Scala:

```
object Solution {
def hasSameDigits(s: String): Boolean = {

}
}
```

Elixir:

```
defmodule Solution do
@spec has_same_digits(s :: String.t) :: boolean
def has_same_digits(s) do

end
end
```

Erlang:

```
-spec has_same_digits(S :: unicode:unicode_binary()) -> boolean().
has_same_digits(S) ->
.
```

Racket:

```
(define/contract (has-same-digits s)
(-> string? boolean?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Check If Digits Are Equal in String After Operations I
* 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:
bool hasSameDigits(string s) {

}
};
```

Java Solution:

```
/**
* Problem: Check If Digits Are Equal in String After Operations I
* 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 boolean hasSameDigits(String s) {

}
}
```

Python3 Solution:

```
"""
Problem: Check If Digits Are Equal in String After Operations I
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 hasSameDigits(self, s: str) -> bool:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def hasSameDigits(self, s):
"""
:type s: str
:rtype: bool
"""
```

JavaScript Solution:

```
/**
* Problem: Check If Digits Are Equal in String After Operations I
* 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} s
* @return {boolean}
*/
var hasSameDigits = function(s) {

};
```

TypeScript Solution:

```
/**
* Problem: Check If Digits Are Equal in String After Operations I
* 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 hasSameDigits(s: string): boolean {

};
```

C# Solution:

```
/*
* Problem: Check If Digits Are Equal in String After Operations I
* 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 bool HasSameDigits(string s) {

}
}
```

C Solution:

```
/*
* Problem: Check If Digits Are Equal in String After Operations I
* 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
```

```
*/

bool hasSameDigits(char* s) {

}
```

Go Solution:

```
// Problem: Check If Digits Are Equal in String After Operations I
// 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 hasSameDigits(s string) bool {

}
```

Kotlin Solution:

```
class Solution {
fun hasSameDigits(s: String): Boolean {

}
}
```

Swift Solution:

```
class Solution {
func hasSameDigits(_ s: String) -> Bool {

}
}
```

Rust Solution:

```
// Problem: Check If Digits Are Equal in String After Operations I
// 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 has_same_digits(s: String) -> bool {

}
}
```

Ruby Solution:

```
# @param {String} s
# @return {Boolean}
def has_same_digits(s)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $s
* @return Boolean
*/
function hasSameDigits($s) {

}
}
```

Dart Solution:

```
class Solution {
bool hasSameDigits(String s) {

}
}
```

Scala Solution:

```
object Solution {
def hasSameDigits(s: String): Boolean = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec has_same_digits(s :: String.t) :: boolean
def has_same_digits(s) do

end
end
```

Erlang Solution:

```
-spec has_same_digits(S :: unicode:unicode_binary()) -> boolean().
has_same_digits(S) ->
.
```

Racket Solution:

```
(define/contract (has-same-digits s)
(-> string? boolean?)
)
```