Problem 2266: Count Number of Texts

Problem Information

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

Problem Description

Alice is texting Bob using her phone. The

mapping

of digits to letters is shown in the figure below.

In order to

add

a letter, Alice has to

press

the key of the corresponding digit

i

times, where

i

is the position of the letter in the key.

For example, to add the letter

's'

, Alice has to press

'7'

four times. Similarly, to add the letter

'k'

, Alice has to press

'5'

twice.

Note that the digits

'0'

and

'1'

do not map to any letters, so Alice

does not

use them.

However, due to an error in transmission, Bob did not receive Alice's text message but received a

string of pressed keys

instead.

For example, when Alice sent the message

"bob"

, Bob received the string

"2266622"

.

Given a string

pressedKeys

representing the string received by Bob, return

the

total number of possible text messages

Alice could have sent

.

Since the answer may be very large, return it

modulo

10

9

+ 7

.

Example 1:

Input:

pressedKeys = "22233"

Output:

8

Explanation:

The possible text messages Alice could have sent are: "aaadd", "abdd", "badd", "cdd", "aaae", "abe", "bae", and "ce". Since there are 8 possible messages, we return 8.

Example 2:

Input:

pressedKeys = "222222222222222222222222222222222222"

Output:

82876089

Explanation:

There are 2082876103 possible text messages Alice could have sent. Since we need to return the answer modulo 10

9

+ 7, we return 2082876103 % (10

9

+ 7) = 82876089.

Constraints:

1 <= pressedKeys.length <= 10

5

pressedKeys

only consists of digits from

'2'

-

'9'

.

Code Snippets

C++:

```
class Solution {
public:
int countTexts(string pressedKeys) {

}
};
```

Java:

```
class Solution {
public int countTexts(String pressedKeys) {

}
}
```

Python3:

```
class Solution:
def countTexts(self, pressedKeys: str) -> int:
```

Python:

```
class Solution(object):
def countTexts(self, pressedKeys):
"""
:type pressedKeys: str
:rtype: int
"""
```

JavaScript:

```
/**
* @param {string} pressedKeys
* @return {number}
*/
var countTexts = function(pressedKeys) {

};
```

TypeScript:

```
function countTexts(pressedKeys: string): number {

};
```

C#:

```
public class Solution {
public int CountTexts(string pressedKeys) {
```

```
}
}
```

C:

```
int countTexts(char* pressedKeys) {

}
```

Go:

```
func countTexts(pressedKeys string) int {

}
```

Kotlin:

```
class Solution {
fun countTexts(pressedKeys: String): Int {

}
}
```

Swift:

```
class Solution {
func countTexts(_ pressedKeys: String) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn count_texts(pressed_keys: String) -> i32 {

}
}
```

Ruby:

```
# @param {String} pressed_keys
# @return {Integer}
def count_texts(pressed_keys)

end
```

PHP:

```
class Solution {

/**
* @param String $pressedKeys
* @return Integer
*/
function countTexts($pressedKeys) {

}
}
```

Dart:

```
class Solution {
int countTexts(String pressedKeys) {

}
}
```

Scala:

```
object Solution {
def countTexts(pressedKeys: String): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec count_texts(pressed_keys :: String.t) :: integer
def count_texts(pressed_keys) do

end
end
```

Erlang:

```
-spec count_texts(PressedKeys :: unicode:unicode_binary()) -> integer().
count_texts(PressedKeys) ->
.
```

Racket:

```
(define/contract (count-texts pressedKeys)
(-> string? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Number of Texts
* Difficulty: Medium
* Tags: string, dp, math, hash
*
* 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:
int countTexts(string pressedKeys) {

}
};
```

Java Solution:

```
/**
* Problem: Count Number of Texts
* Difficulty: Medium
* Tags: string, dp, math, hash
*
* 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 int countTexts(String pressedKeys) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Number of Texts
Difficulty: Medium
Tags: string, dp, math, hash

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 countTexts(self, pressedKeys: str) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def countTexts(self, pressedKeys):
"""
:type pressedKeys: str
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Count Number of Texts
* Difficulty: Medium
```

```
* Tags: string, dp, math, hash
*
* 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} pressedKeys
* @return {number}
*/
var countTexts = function(pressedKeys) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Number of Texts
* Difficulty: Medium
* Tags: string, dp, math, hash
*
* 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 countTexts(pressedKeys: string): number {

};
```

C# Solution:

```
/*
* Problem: Count Number of Texts
* Difficulty: Medium
* Tags: string, dp, math, hash
*
* 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 int CountTexts(string pressedKeys) {

}
}
```

C Solution:

```
/*
* Problem: Count Number of Texts
* Difficulty: Medium
* Tags: string, dp, math, hash
*
* 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
*/

int countTexts(char* pressedKeys) {

}
```

Go Solution:

```
// Problem: Count Number of Texts
// Difficulty: Medium
// Tags: string, dp, math, hash
//
// 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 countTexts(pressedKeys string) int {

}
```

Kotlin Solution:

```
class Solution {
fun countTexts(pressedKeys: String): Int {

}
}
```

Swift Solution:

```
class Solution {
func countTexts(_ pressedKeys: String) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Number of Texts
// Difficulty: Medium
// Tags: string, dp, math, hash
//
// 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 count_texts(pressed_keys: String) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String} pressed_keys
# @return {Integer}
def count_texts(pressed_keys)

end
```

PHP Solution:

```
class Solution {
```

```
/**
* @param String $pressedKeys
* @return Integer
*/
function countTexts($pressedKeys) {

}
}
```

Dart Solution:

```
class Solution {
int countTexts(String pressedKeys) {

}
}
```

Scala Solution:

```
object Solution {
def countTexts(pressedKeys: String): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec count_texts(pressed_keys :: String.t) :: integer
def count_texts(pressed_keys) do

end
end
```

Erlang Solution:

```
-spec count_texts(PressedKeys :: unicode:unicode_binary()) -> integer().
count_texts(PressedKeys) ->
.
```

Racket Solution:

```
(define/contract (count-texts pressedKeys)
(-> string? exact-integer?)
)
```