Problem 389: Find the Difference

Problem Information

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

Problem Description

You are given two strings

s

and

t

.

String

t

is generated by random shuffling string

s

and then add one more letter at a random position.

Return the letter that was added to

t

.

Example 1:

Input:

s = "abcd", t = "abcde"

Output:

"e"

Explanation:

'e' is the letter that was added.

Example 2:

Input:

s = "", t = "y"

Output:

"y"

Constraints:

0 <= s.length <= 1000

t.length == s.length + 1

s

and

t

consist of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
char findTheDifference(string s, string t) {

}
};
```

Java:

```
class Solution {
public char findTheDifference(String s, String t) {

}
}
```

Python3:

```
class Solution:
def findTheDifference(self, s: str, t: str) -> str:
```

Python:

```
class Solution(object):
def findTheDifference(self, s, t):
"""
:type s: str
:type t: str
:rtype: str
"""
```

JavaScript:

```
/**
* @param {string} s
* @param {string} t
* @return {character}
*/
var findTheDifference = function(s, t) {
```

```
};
```

TypeScript:

```
function findTheDifference(s: string, t: string): string {

};
```

C#:

```
public class Solution {
public char FindTheDifference(string s, string t) {

}
}
```

C:

```
char findTheDifference(char* s, char* t) {

}
```

Go:

```
func findTheDifference(s string, t string) byte {

}
```

Kotlin:

```
class Solution {
fun findTheDifference(s: String, t: String): Char {

}
}
```

Swift:

```
class Solution {
func findTheDifference(_ s: String, _ t: String) -> Character {

}
```

```
}
```

Rust:

```
impl Solution {
pub fn find_the_difference(s: String, t: String) -> char {

}
}
```

Ruby:

```
# @param {String} s
# @param {String} t
# @return {Character}
def find_the_difference(s, t)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @param String $t
* @return String
*/
function findTheDifference($s, $t) {

}
}
```

Dart:

```
class Solution {
String findTheDifference(String s, String t) {

}
}
```

Scala:

```
object Solution {
def findTheDifference(s: String, t: String): Char = {

}
}
```

Elixir:

```
defmodule Solution do
@spec find_the_difference(s :: String.t, t :: String.t) :: char
def find_the_difference(s, t) do

end
end
```

Erlang:

```
-spec find_the_difference(S :: unicode:unicode_binary(), T ::
unicode:unicode_binary()) -> char().
find_the_difference(S, T) ->
.
```

Racket:

```
(define/contract (find-the-difference s t)
(-> string? string? char?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/
```

```
class Solution {
public:
char findTheDifference(string s, string t) {

}
};
```

Java Solution:

```
/**
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

class Solution {
public char findTheDifference(String s, String t) {

}
}
```

Python3 Solution:

```
"""
Problem: Find the Difference
Difficulty: Easy
Tags: string, hash, sort

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

class Solution:
def findTheDifference(self, s: str, t: str) -> str:
# TODO: Implement optimized solution
```

```
pass
```

Python Solution:

```
class Solution(object):
def findTheDifference(self, s, t):
"""
:type s: str
:type t: str
:rtype: str
"""
```

JavaScript Solution:

```
/**
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

/**
* @param {string} s
* @param {string} t
* @return {character}
*/
var findTheDifference = function(s, t) {

};
```

TypeScript Solution:

```
/**
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
```

```
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

function findTheDifference(s: string, t: string): string {

};
```

C# Solution:

```
/*
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

public class Solution {
public char FindTheDifference(string s, string t) {

}
}
```

C Solution:

```
/*
* Problem: Find the Difference
* Difficulty: Easy
* Tags: string, hash, sort
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

char findTheDifference(char* s, char* t) {

}
```

Go Solution:

```
// Problem: Find the Difference
// Difficulty: Easy
// Tags: string, hash, sort
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

func findTheDifference(s string, t string) byte {

}
```

Kotlin Solution:

```
class Solution {
fun findTheDifference(s: String, t: String): Char {

}
}
```

Swift Solution:

```
class Solution {
func findTheDifference(_ s: String, _ t: String) -> Character {

}
}
```

Rust Solution:

```
// Problem: Find the Difference
// Difficulty: Easy
// Tags: string, hash, sort
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

impl Solution {
pub fn find_the_difference(s: String, t: String) -> char {
```

```
}
}
```

Ruby Solution:

```
# @param {String} s
# @param {String} t
# @return {Character}
def find_the_difference(s, t)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $s
* @param String $t
* @return String
*/
function findTheDifference($s, $t) {

}
}
```

Dart Solution:

```
class Solution {
String findTheDifference(String s, String t) {

}
}
```

Scala Solution:

```
object Solution {
def findTheDifference(s: String, t: String): Char = {

}
```

```
}
```

Elixir Solution:

```
defmodule Solution do
@spec find_the_difference(s :: String.t, t :: String.t) :: char
def find_the_difference(s, t) do

end
end
```

Erlang Solution:

```
-spec find_the_difference(S :: unicode:unicode_binary(), T ::
unicode:unicode_binary()) -> char().
find_the_difference(S, T) ->
.
```

Racket Solution:

```
(define/contract (find-the-difference s t)
(-> string? string? char?)
)
```