Problem 3114: Latest Time You Can Obtain After Replacing Characters

Problem Information

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

Problem Description

You are given a string

s

representing a 12-hour format time where some of the digits (possibly none) are replaced with a

"?"

.

12-hour times are formatted as

"HH:MM"

, where

HH

is between

00

and

11

, and

MM

is between

00

and

59

. The earliest 12-hour time is

00:00

, and the latest is

11:59

.

You have to replace

all

the

"?"

characters in

s

with digits such that the time we obtain by the resulting string is a

valid

12-hour format time and is the

latest

possible.

Return

the resulting string

.

Example 1:

Input:

s = "1?:?4"

Output:

"11:54"

Explanation:

The latest 12-hour format time we can achieve by replacing

"?"

characters is

"11:54"

.

Example 2:

Input:

s = "0?:5?"

Output:

"09:59"

Explanation:

The latest 12-hour format time we can achieve by replacing

"?"

characters is

"09:59"

.

Constraints:

s.length == 5

s[2]

is equal to the character

":"

.

All characters except

s[2]

are digits or

"?"

characters.

The input is generated such that there is

at least

one time between

"00:00"

and

"11:59"

that you can obtain after replacing the

"?"

characters.

Code Snippets

C++:

```
class Solution {
public:
string findLatestTime(string s) {

}
};
```

Java:

```
class Solution {
public String findLatestTime(String s) {

}
}
```

Python3:

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

Python:

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

JavaScript:

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

};
```

TypeScript:

```
function findLatestTime(s: string): string {

};
```

C#:

```
public class Solution {
public string FindLatestTime(string s) {

}
}
```

C:

```
char* findLatestTime(char* s) {

}
```

Go:

```
func findLatestTime(s string) string {

}
```

Kotlin:

```
class Solution {
fun findLatestTime(s: String): String {

}
}
```

Swift:

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

}
}
```

Rust:

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

}
}
```

Ruby:

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

end
```

PHP:

```
class Solution {

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

}
}
```

Dart:

```
class Solution {
String findLatestTime(String s) {

}
}
```

Scala:

```
object Solution {
def findLatestTime(s: String): String = {

}
}
```

Elixir:

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

end
end
```

Erlang:

```
-spec find_latest_time(S :: unicode:unicode_binary()) ->
unicode:unicode_binary().
find_latest_time(S) ->
.
```

Racket:

```
(define/contract (find-latest-time s)
(-> string? string?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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 findLatestTime(string s) {

}
};
```

Java Solution:

```
/**
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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 findLatestTime(String s) {

}
}
```

Python3 Solution:

```
"""
Problem: Latest Time You Can Obtain After Replacing Characters
Difficulty: Easy
Tags: string

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 findLatestTime(self, s: str) -> str:
# TODO: Implement optimized solution
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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 {string}
*/
```

```
var findLatestTime = function(s) {

};
```

TypeScript Solution:

```
/**
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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 findLatestTime(s: string): string {

};
```

C# Solution:

```
/*
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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 FindLatestTime(string s) {

}
}
```

C Solution:

```
/*
* Problem: Latest Time You Can Obtain After Replacing Characters
* Difficulty: Easy
* Tags: string
*
* 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* findLatestTime(char* s) {

}
```

Go Solution:

```
// Problem: Latest Time You Can Obtain After Replacing Characters
// Difficulty: Easy
// Tags: string
//
// 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 findLatestTime(s string) string {

}
```

Kotlin Solution:

```
class Solution {
fun findLatestTime(s: String): String {

}
}
```

Swift Solution:

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

}
}
```

Rust Solution:

```
// Problem: Latest Time You Can Obtain After Replacing Characters
// Difficulty: Easy
// Tags: string
//
// 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 find_latest_time(s: String) -> String {

}
}
```

Ruby Solution:

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

end
```

PHP Solution:

```
class Solution {

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

}
}
```

Dart Solution:

```
class Solution {
String findLatestTime(String s) {

}
}
```

Scala Solution:

```
object Solution {
def findLatestTime(s: String): String = {

}
}
```

Elixir Solution:

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

end
end
```

Erlang Solution:

```
-spec find_latest_time(S :: unicode:unicode_binary()) ->
unicode:unicode_binary().
find_latest_time(S) ->
.
```

Racket Solution:

```
(define/contract (find-latest-time s)
(-> string? string?)
)
```