Problem 539: Minimum Time Difference

Problem Information

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

Problem Description

Given a list of 24-hour clock time points in

"HH:MM"

format, return

the minimum

minutes

difference between any two time-points in the list

.

Example 1:

Input:

timePoints = ["23:59","00:00"]

Output:

1

Example 2:

Input:

timePoints = ["00:00","23:59","00:00"]

Output:

0

Constraints:

2 <= timePoints.length <= 2 * 10

4

timePoints[i]

is in the format

"HH:MM"

.

Code Snippets

C++:

```
class Solution {
public:
int findMinDifference(vector<string>& timePoints) {

}
};
```

Java:

```
class Solution {
public int findMinDifference(List<String> timePoints) {

}
```

```
}
```

Python3:

```
class Solution:
def findMinDifference(self, timePoints: List[str]) -> int:
```

Python:

```
class Solution(object):
def findMinDifference(self, timePoints):
"""
:type timePoints: List[str]
:rtype: int
"""
```

JavaScript:

```
/**
* @param {string[]} timePoints
* @return {number}
*/
var findMinDifference = function(timePoints) {

};
```

TypeScript:

```
function findMinDifference(timePoints: string[]): number {

};
```

C#:

```
public class Solution {
public int FindMinDifference(IList<string> timePoints) {

}
}
```

C:

```
int findMinDifference(char** timePoints, int timePointsSize) {

}
```

Go:

```
func findMinDifference(timePoints []string) int {

}
```

Kotlin:

```
class Solution {
fun findMinDifference(timePoints: List<String>): Int {

}
}
```

Swift:

```
class Solution {
func findMinDifference(_ timePoints: [String]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn find_min_difference(time_points: Vec<String>) -> i32 {

}
}
```

Ruby:

```
# @param {String[]} time_points
# @return {Integer}
def find_min_difference(time_points)

end
```

PHP:

```
class Solution {

/**
* @param String[] $timePoints
* @return Integer
*/
function findMinDifference($timePoints) {

}
}
```

Dart:

```
class Solution {
int findMinDifference(List<String> timePoints) {

}
}
```

Scala:

```
object Solution {
def findMinDifference(timePoints: List[String]): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec find_min_difference(time_points :: [String.t]) :: integer
def find_min_difference(time_points) do

end
end
```

Erlang:

```
-spec find_min_difference(TimePoints :: [unicode:unicode_binary()]) ->
integer().
find_min_difference(TimePoints) ->
.
```

Racket:

```
(define/contract (find-min-difference timePoints)
(-> (listof string?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
int findMinDifference(vector<string>& timePoints) {

}
};
```

Java Solution:

```
/**
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int findMinDifference(List<String> timePoints) {
```

```
}
}
```

Python3 Solution:

```
"""
Problem: Minimum Time Difference
Difficulty: Medium
Tags: array, string, math, sort

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

class Solution:
def findMinDifference(self, timePoints: List[str]) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def findMinDifference(self, timePoints):
"""
:type timePoints: List[str]
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/
```

```
/**
* @param {string[]} timePoints
* @return {number}
*/
var findMinDifference = function(timePoints) {

};
```

TypeScript Solution:

```
/**
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function findMinDifference(timePoints: string[]): number {

};
```

C# Solution:

```
/*
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int FindMinDifference(IList<string> timePoints) {

}
```

```
}
```

C Solution:

```
/*
* Problem: Minimum Time Difference
* Difficulty: Medium
* Tags: array, string, math, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

int findMinDifference(char** timePoints, int timePointsSize) {

}
```

Go Solution:

```
// Problem: Minimum Time Difference
// Difficulty: Medium
// Tags: array, string, math, sort
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func findMinDifference(timePoints []string) int {

}
```

Kotlin Solution:

```
class Solution {
fun findMinDifference(timePoints: List<String>): Int {

}
}
```

Swift Solution:

```
class Solution {
func findMinDifference(_ timePoints: [String]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Minimum Time Difference
// Difficulty: Medium
// Tags: array, string, math, sort
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn find_min_difference(time_points: Vec<String>) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String[]} time_points
# @return {Integer}
def find_min_difference(time_points)

end
```

PHP Solution:

```
class Solution {

/**
* @param String[] $timePoints
* @return Integer
*/
function findMinDifference($timePoints) {

}
}
```

Dart Solution:

```
class Solution {
int findMinDifference(List<String> timePoints) {

}
}
```

Scala Solution:

```
object Solution {
def findMinDifference(timePoints: List[String]): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec find_min_difference(time_points :: [String.t]) :: integer
def find_min_difference(time_points) do

end
end
```

Erlang Solution:

```
-spec find_min_difference(TimePoints :: [unicode:unicode_binary()]) ->
integer().
find_min_difference(TimePoints) ->
.
```

Racket Solution:

```
(define/contract (find-min-difference timePoints)
(-> (listof string?) exact-integer?)
)
```