Problem 128: Longest Consecutive Sequence

Problem Information

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

Problem Description

Given an unsorted array of integers

nums

, return

the length of the longest consecutive elements sequence.

You must write an algorithm that runs in

O(n)

time.

Example 1:

Input:

nums = [100,4,200,1,3,2]

Output:

4

Explanation:

The longest consecutive elements sequence is

[1, 2, 3, 4]

. Therefore its length is 4.

Example 2:

Input:

nums = [0,3,7,2,5,8,4,6,0,1]

Output:

9

Example 3:

Input:

nums = [1,0,1,2]

Output:

3

Constraints:

0 <= nums.length <= 10

5

-10

9

<= nums[i] <= 10

9

Code Snippets

C++:

```
class Solution {
public:
int longestConsecutive(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int longestConsecutive(int[] nums) {

}
}
```

Python3:

```
class Solution:
def longestConsecutive(self, nums: List[int]) -> int:
```

Python:

```
class Solution(object):
def longestConsecutive(self, nums):
"""
:type nums: List[int]
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number[]} nums
* @return {number}
*/
var longestConsecutive = function(nums) {

};
```

TypeScript:

```
function longestConsecutive(nums: number[]): number {

};
```

C#:

```
public class Solution {
public int LongestConsecutive(int[] nums) {

}
}
```

C:

```
int longestConsecutive(int* nums, int numsSize) {

}
```

Go:

```
func longestConsecutive(nums []int) int {

}
```

Kotlin:

```
class Solution {
fun longestConsecutive(nums: IntArray): Int {

}
}
```

Swift:

```
class Solution {
func longestConsecutive(_ nums: [Int]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn longest_consecutive(nums: Vec<i32>) -> i32 {

}
}
```

Ruby:

```
# @param {Integer[]} nums
# @return {Integer}
def longest_consecutive(nums)

end
```

PHP:

```
class Solution {

/**
* @param Integer[] $nums
* @return Integer
*/
function longestConsecutive($nums) {

}
}
```

Dart:

```
class Solution {
int longestConsecutive(List<int> nums) {

}
}
```

Scala:

```
object Solution {
def longestConsecutive(nums: Array[Int]): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec longest_consecutive(nums :: [integer]) :: integer
def longest_consecutive(nums) do

end
end
```

Erlang:

```
-spec longest_consecutive(Nums :: [integer()]) -> integer().
longest_consecutive(Nums) ->
.
```

Racket:

```
(define/contract (longest-consecutive nums)
(-> (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/
```

```
class Solution {
public:
int longestConsecutive(vector<int>& nums) {

}
};
```

Java Solution:

```
/**
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

class Solution {
public int longestConsecutive(int[] nums) {

}
}
```

Python3 Solution:

```
"""
Problem: Longest Consecutive Sequence
Difficulty: Medium
Tags: array, graph, hash, sort

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) for hash map
"""

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

Python Solution:

```
class Solution(object):
def longestConsecutive(self, nums):
"""
:type nums: List[int]
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

/**
* @param {number[]} nums
* @return {number}
*/
var longestConsecutive = function(nums) {

};
```

TypeScript Solution:

```
/**
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

function longestConsecutive(nums: number[]): number {
};
```

C# Solution:

```
/*
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

public class Solution {
public int LongestConsecutive(int[] nums) {

}
}
```

C Solution:

```
/*
* Problem: Longest Consecutive Sequence
* Difficulty: Medium
* Tags: array, graph, hash, sort
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

int longestConsecutive(int* nums, int numsSize) {

}
```

Go Solution:

```
// Problem: Longest Consecutive Sequence
// Difficulty: Medium
// Tags: array, graph, hash, sort
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

func longestConsecutive(nums []int) int {

}
```

Kotlin Solution:

```
class Solution {
fun longestConsecutive(nums: IntArray): Int {

}
}
```

Swift Solution:

```
class Solution {
func longestConsecutive(_ nums: [Int]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Longest Consecutive Sequence
// Difficulty: Medium
// Tags: array, graph, hash, sort
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

impl Solution {
pub fn longest_consecutive(nums: Vec<i32>) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[]} nums
# @return {Integer}
def longest_consecutive(nums)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[] $nums
* @return Integer
*/
function longestConsecutive($nums) {

}
}
```

Dart Solution:

```
class Solution {
int longestConsecutive(List<int> nums) {

}
}
```

Scala Solution:

```
object Solution {
def longestConsecutive(nums: Array[Int]): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec longest_consecutive(nums :: [integer]) :: integer
def longest_consecutive(nums) do
```

```
end
end
```

Erlang Solution:

```
-spec longest_consecutive(Nums :: [integer()]) -> integer().
longest_consecutive(Nums) ->
.
```

Racket Solution:

```
(define/contract (longest-consecutive nums)
(-> (listof exact-integer?) exact-integer?)
)
```