Problem 702: Search in a Sorted Array of Unknown Size

Problem Information

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

Problem Description

This is an

interactive problem

.

You have a sorted array of

unique

elements and an

unknown size

. You do not have an access to the array but you can use the

ArrayReader

interface to access it. You can call

ArrayReader.get(i)

that:

returns the value at the

i

th

index (

0-indexed

) of the secret array (i.e.,

secret[i]

), or

returns

2

31

- 1

if the

i

is out of the boundary of the array.

You are also given an integer

target

.

Return the index

k

of the hidden array where

secret[k] == target

or return

-1

otherwise.

You must write an algorithm with

O(log n)

runtime complexity.

Example 1:

Input:

secret = [-1,0,3,5,9,12], target = 9

Output:

4

Explanation:

9 exists in secret and its index is 4.

Example 2:

Input:

secret = [-1,0,3,5,9,12], target = 2

Output:

-1

Explanation:

2 does not exist in secret so return -1.

Constraints:

1 <= secret.length <= 10

4

-10

4

<= secret[i], target <= 10

4

secret

is sorted in a strictly increasing order.

Code Snippets

C++:

```
/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* public:
* int get(int index);
* };
*/

class Solution {
public:
int search(const ArrayReader& reader, int target) {
```

```
}
};
```

Java:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* interface ArrayReader {
* public int get(int index) {}
* }
*/

class Solution {
public int search(ArrayReader reader, int target) {

}
}
```

Python3:

```
# """
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# """
#class ArrayReader:
# def get(self, index: int) -> int:

class Solution:
def search(self, reader: 'ArrayReader', target: int) -> int:
```

Python:

```
# """
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# """
#class ArrayReader(object):
# def get(self, index):
# """
# :type index: int
# :rtype int
```

```
# """

class Solution(object):
def search(self, reader, target):
"""
:type reader: ArrayReader
:type target: int
:rtype: int
"""
```

JavaScript:

```
/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* function ArrayReader() {
*
* @param {number} index
* @return {number}
* this.get = function(index) {
* ...
* };
* };
*/

/**
* @param {ArrayReader} reader
* @param {number} target
* @return {number}
*/
var search = function (reader, target) {

};
```

TypeScript:

```
/**
* class ArrayReader {
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* get(index: number): number {};
* };
```

```
*/

function search(reader: ArrayReader, target: number): number {

};
```

C#:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* public int Get(int index) {}
* }
*/

class Solution {
public int Search(ArrayReader reader, int target) {

}
}
```

C:

```
/**
* ********************************************************************
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* ********************************************************************
*
* int getElement(ArrayReader *, int index);
*/

int search(struct ArrayReader* reader, int target) {

}
```

Go:

```
/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
```

```
* type ArrayReader struct {
* }
*
* func (this *ArrayReader) get(index int) int {}
*/

func search(reader ArrayReader, target int) int {

}
```

Kotlin:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* fun get(index: Int): Int {}
* }
*/

class Solution {
fun search(reader: ArrayReader, target: Int): Int {

}
}
```

Swift:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* public class ArrayReader {
* public func get(_ index: Int) -> Int {}
* }
*/

class Solution {
func search(_ reader: ArrayReader, _ target: Int) -> Int {

}
}
```

Ruby:

```
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# class ArrayReader
# def get(index)
#
# end
# end

# @param {ArrayReader} reader
# @param {int} target
# @return {int}
def search(reader, target)

end
```

PHP:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* function get($index) {}
* }
*/

class Solution {
/**
* @param ArrayReader $reader
* @param Integer $target
* @return Integer
*/
function search($reader, $target) {

}
}
```

Scala:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
```

```
* class ArrayReader {
* def get(index: Int): Int = {}
* }
*/

object Solution {
def search(reader: ArrayReader, target: Int): Int = {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
* 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
*/

/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* public:
* int get(int index);
* };
*/

class Solution {
public:
int search(const ArrayReader& reader, int target) {

}
};
```

Java Solution:

```
/**
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
* 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
*/

/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* interface ArrayReader {
* public int get(int index) {
// TODO: Implement optimized solution
return 0;
}
* }
*/

class Solution {
public int search(ArrayReader reader, int target) {

}
}
```

Python3 Solution:

```
"""
Problem: Search in a Sorted Array of Unknown Size
Difficulty: Medium
Tags: array, sort, search

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
"""

# """
```

```
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# """
#class ArrayReader:
# def get(self, index: int) -> int:

class Solution:
def search(self, reader: 'ArrayReader', target: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
# """
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# """
#class ArrayReader(object):
# def get(self, index):
# """
# :type index: int
# :rtype int
# """

class Solution(object):
def search(self, reader, target):
"""
:type reader: ArrayReader
:type target: int
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
* 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
*/

/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* function ArrayReader() {
*
* @param {number} index
* @return {number}
* this.get = function(index) {
* ...
* };
* };
*/

/**
* @param {ArrayReader} reader
* @param {number} target
* @return {number}
*/
var search = function (reader, target) {

};
```

TypeScript Solution:

```
/**
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
* 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 ArrayReader {
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
```

```
* get(index: number): number {};
* };
*/

function search(reader: ArrayReader, target: number): number {

};
```

C# Solution:

```
/*
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
* 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
*/

/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* public int Get(int index) {}
* }
*/

class Solution {
public int Search(ArrayReader reader, int target) {

}
}
```

C Solution:

```
/*
* Problem: Search in a Sorted Array of Unknown Size
* Difficulty: Medium
* Tags: array, sort, search
*
```

```
* 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
*/

/**
* ****************************************************************
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* ****************************************************************
*
* int getElement(ArrayReader *, int index);
*/

int search(struct ArrayReader* reader, int target) {

}
```

Go Solution:

```
// Problem: Search in a Sorted Array of Unknown Size
// Difficulty: Medium
// Tags: array, sort, search
//
// 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

/**
* // This is the ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* type ArrayReader struct {
* }
*
* func (this *ArrayReader) get(index int) int {}
*/

func search(reader ArrayReader, target int) int {

}
```

Kotlin Solution:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* fun get(index: Int): Int {}
* }
*/

class Solution {
fun search(reader: ArrayReader, target: Int): Int {

}
}
```

Swift Solution:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* public class ArrayReader {
* public func get(_ index: Int) -> Int {}
* }
*/

class Solution {
func search(_ reader: ArrayReader, _ target: Int) -> Int {

}
}
```

Ruby Solution:

```
# This is ArrayReader's API interface.
# You should not implement it, or speculate about its implementation
# class ArrayReader
# def get(index)
#
# end
# end

# @param {ArrayReader} reader
```

```
# @param {int} target
# @return {int}
def search(reader, target)

end
```

PHP Solution:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* function get($index) {}
* }
*/

class Solution {
/**
* @param ArrayReader $reader
* @param Integer $target
* @return Integer
*/
function search($reader, $target) {

}
}
```

Scala Solution:

```
/**
* // This is ArrayReader's API interface.
* // You should not implement it, or speculate about its implementation
* class ArrayReader {
* def get(index: Int): Int = {}
* }
*/

object Solution {
def search(reader: ArrayReader, target: Int): Int = {

}
```

```
}
```