Problem 1953: Maximum Number of Weeks for Which You Can Work

Problem Information

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

Problem Description

There are

n

projects numbered from

0

to

n - 1

. You are given an integer array

milestones

where each

milestones[i]

denotes the number of milestones the

i

th

project has.

You can work on the projects following these two rules:

Every week, you will finish

exactly one

milestone of

one

project. You

must

work every week.

You

cannot

work on two milestones from the same project for two

consecutive

weeks.

Once all the milestones of all the projects are finished, or if the only milestones that you can work on will cause you to violate the above rules, you will

stop working

. Note that you may not be able to finish every project's milestones due to these constraints.

Return

the

maximum

number of weeks you would be able to work on the projects without violating the rules mentioned above

.

Example 1:

Input:

milestones = [1,2,3]

Output:

6

Explanation:

One possible scenario is: - During the 1

st

week, you will work on a milestone of project 0. - During the 2

nd

week, you will work on a milestone of project 2. - During the 3

rd

week, you will work on a milestone of project 1. - During the 4

th

week, you will work on a milestone of project 2. - During the 5

th

week, you will work on a milestone of project 1. - During the 6

th

week, you will work on a milestone of project 2. The total number of weeks is 6.

Example 2:

Input:

milestones = [5,2,1]

Output:

7

Explanation:

One possible scenario is: - During the 1

st

week, you will work on a milestone of project 0. - During the 2

nd

week, you will work on a milestone of project 1. - During the 3

rd

week, you will work on a milestone of project 0. - During the 4

th

week, you will work on a milestone of project 1. - During the 5

th

week, you will work on a milestone of project 0. - During the 6

th

week, you will work on a milestone of project 2. - During the 7

th

week, you will work on a milestone of project 0. The total number of weeks is 7. Note that you cannot work on the last milestone of project 0 on 8

th

week because it would violate the rules. Thus, one milestone in project 0 will remain unfinished.

Constraints:

n == milestones.length

1 <= n <= 10

5

1 <= milestones[i] <= 10

9

Code Snippets

C++:

```
class Solution {
public:
long long numberOfWeeks(vector<int>& milestones) {

}
};
```

Java:

```
class Solution {
public long numberOfWeeks(int[] milestones) {

}
}
```

Python3:

```
class Solution:
def numberOfWeeks(self, milestones: List[int]) -> int:
```

Python:

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

JavaScript:

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

};
```

TypeScript:

```
function numberOfWeeks(milestones: number[]): number {

};
```

C#:

```
public class Solution {
public long NumberOfWeeks(int[] milestones) {

}
}
```

C:

```
long long numberOfWeeks(int* milestones, int milestonesSize) {

}
```

Go:

```
func numberOfWeeks(milestones []int) int64 {

}
```

Kotlin:

```
class Solution {
fun numberOfWeeks(milestones: IntArray): Long {

}
}
```

Swift:

```
class Solution {
func numberOfWeeks(_ milestones: [Int]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn number_of_weeks(milestones: Vec<i32>) -> i64 {

}
}
```

Ruby:

```
# @param {Integer[]} milestones
# @return {Integer}
def number_of_weeks(milestones)

end
```

PHP:

```
class Solution {

/**
* @param Integer[] $milestones
* @return Integer
*/
function numberOfWeeks($milestones) {

}
}
```

Dart:

```
class Solution {
int numberOfWeeks(List<int> milestones) {

}
}
```

Scala:

```
object Solution {
def numberOfWeeks(milestones: Array[Int]): Long = {

}
}
```

Elixir:

```
defmodule Solution do
@spec number_of_weeks(milestones :: [integer]) :: integer
def number_of_weeks(milestones) do

end
end
```

Erlang:

```
-spec number_of_weeks(Milestones :: [integer()]) -> integer().
number_of_weeks(Milestones) ->
.
```

Racket:

```
(define/contract (number-of-weeks milestones)
(-> (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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:
long long numberOfWeeks(vector<int>& milestones) {

}
};
```

Java Solution:

```
/**
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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 long numberOfWeeks(int[] milestones) {

}
}
```

Python3 Solution:

```
"""
Problem: Maximum Number of Weeks for Which You Can Work
Difficulty: Medium
Tags: array, greedy

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 numberOfWeeks(self, milestones: List[int]) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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 {number[]} milestones
* @return {number}
*/
var numberOfWeeks = function(milestones) {

};
```

TypeScript Solution:

```
/**
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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 numberOfWeeks(milestones: number[]): number {

};
```

C# Solution:

```
/*
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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 long NumberOfWeeks(int[] milestones) {

}
}
```

C Solution:

```
/*
* Problem: Maximum Number of Weeks for Which You Can Work
* Difficulty: Medium
* Tags: array, greedy
*
* 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
*/

long long numberOfWeeks(int* milestones, int milestonesSize) {

}
```

Go Solution:

```
// Problem: Maximum Number of Weeks for Which You Can Work
// Difficulty: Medium
// Tags: array, greedy
//
// 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 numberOfWeeks(milestones []int) int64 {

}
```

Kotlin Solution:

```
class Solution {
fun numberOfWeeks(milestones: IntArray): Long {

}
}
```

Swift Solution:

```
class Solution {
func numberOfWeeks(_ milestones: [Int]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Maximum Number of Weeks for Which You Can Work
// Difficulty: Medium
// Tags: array, greedy
//
// 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 number_of_weeks(milestones: Vec<i32>) -> i64 {

}
}
```

Ruby Solution:

```
# @param {Integer[]} milestones
# @return {Integer}
def number_of_weeks(milestones)

end
```

PHP Solution:

```
class Solution {
```

```
/**
* @param Integer[] $milestones
* @return Integer
*/
function numberOfWeeks($milestones) {

}
}
```

Dart Solution:

```
class Solution {
int numberOfWeeks(List<int> milestones) {

}
}
```

Scala Solution:

```
object Solution {
def numberOfWeeks(milestones: Array[Int]): Long = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec number_of_weeks(milestones :: [integer]) :: integer
def number_of_weeks(milestones) do

end
end
```

Erlang Solution:

```
-spec number_of_weeks(Milestones :: [integer()]) -> integer().
number_of_weeks(Milestones) ->
.
```

Racket Solution:

```
(define/contract (number-of-weeks milestones)
(-> (listof exact-integer?) exact-integer?)
)
```