Problem 2649: Nested Array Generator

Problem Information

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

Problem Description

Given a

multi-dimensional array

of integers, return a generator object which yields integers in the same order as

inorder traversal

.

A

multi-dimensional array

is a recursive data structure that contains both integers and other

multi-dimensional arrays

.

inorder traversal

iterates over each array from left to right, yielding any integers it encounters or applying

inorder traversal

to any arrays it encounters.

Example 1:

Input:

arr = [[[6]],[1,3],[]]

Output:

[6,1,3]

Explanation:

const generator = inorderTraversal(arr); generator.next().value; // 6 generator.next().value; // 1 generator.next().value; // 3 generator.next().done; // true

Example 2:

Input:

arr = []

Output:

[]

Explanation:

There are no integers so the generator doesn't yield anything.

Constraints:

0 <= arr.flat().length <= 10

5

0 <= arr.flat()[i] <= 10

5

maxNestingDepth <= 10

5

Can you solve this without creating a new flattened version of the array?

Code Snippets

JavaScript:

```
/**
* @param {Array} arr
* @return {Generator}
*/
var inorderTraversal = function*(arr) {

};

/**
* const gen = inorderTraversal([1, [2, 3]]);
* gen.next().value; // 1
* gen.next().value; // 2
* gen.next().value; // 3
*/
```

TypeScript:

```
type MultidimensionalArray = (MultidimensionalArray | number)[]

function* inorderTraversal(arr: MultidimensionalArray): Generator<number,
void, unknown> {

};

/**
* const gen = inorderTraversal([1, [2, 3]]);
* gen.next().value; // 1
* gen.next().value; // 2
* gen.next().value; // 3
```

```
*/
```

Solutions

JavaScript Solution:

```
/**
* Problem: Nested Array Generator
* Difficulty: Medium
* Tags: array
*
* 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 {Array} arr
* @return {Generator}
*/
var inorderTraversal = function*(arr) {

};

/**
* const gen = inorderTraversal([1, [2, 3]]);
* gen.next().value; // 1
* gen.next().value; // 2
* gen.next().value; // 3
*/
```

TypeScript Solution:

```
/**
* Problem: Nested Array Generator
* Difficulty: Medium
* Tags: array
*
* 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
*/

type MultidimensionalArray = (MultidimensionalArray | number)[]

function* inorderTraversal(arr: MultidimensionalArray): Generator<number,
void, unknown> {

};

/**
* const gen = inorderTraversal([1, [2, 3]]);
* gen.next().value; // 1
* gen.next().value; // 2
* gen.next().value; // 3
*/
```