Problem 2635: Apply Transform Over Each Element in Array

Problem Information

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

Problem Description

Given an integer array

arr

and a mapping function

fn

, return a new array with a transformation applied to each element.

The returned array should be created such that

returnedArray[i] = fn(arr[i], i)

.

Please solve it without the built-in

Array.map

method.

Example 1:

Input:

arr = [1,2,3], fn = function plusone(n) { return n + 1; }

Output:

[2,3,4]

Explanation:

const newArray = map(arr, plusone); // [2,3,4] The function increases each value in the array by one.

Example 2:

Input:

arr = [1,2,3], fn = function plusI(n, i) { return n + i; }

Output:

[1,3,5]

Explanation:

The function increases each value by the index it resides in.

Example 3:

Input:

arr = [10,20,30], fn = function constant() { return 42; }

Output:

[42,42,42]

Explanation:

The function always returns 42.

Constraints:

0 <= arr.length <= 1000

-10

9

<= arr[i] <= 10

9

fn

returns an integer.

Code Snippets

JavaScript:

```
/**
* @param {number[]} arr
* @param {Function} fn
* @return {number[]}
*/
var map = function(arr, fn) {

};
```

TypeScript:

```
function map(arr: number[], fn: (n: number, i: number) => number): number[] {

};
```

Solutions

JavaScript Solution:

```
/**
* Problem: Apply Transform Over Each Element in Array
* Difficulty: Easy
* 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 {number[]} arr
* @param {Function} fn
* @return {number[]}
*/
var map = function(arr, fn) {

};
```

TypeScript Solution:

```
/**
* Problem: Apply Transform Over Each Element in Array
* Difficulty: Easy
* 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
*/

function map(arr: number[], fn: (n: number, i: number) => number): number[] {

};
```