Problem 2804: Array Prototype ForEach

Problem Information

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

Problem Description

Write your version of method

forEach

that enhances all arrays such that you can call the

array.forEach(callback, context)

method on any array and it will execute

callback

on each element of the array. Method

forEach

should not return anything.

callback

accepts the following arguments:

currentValue

- represents the current element being processed in the array. It is the value of the element in the current iteration.

index

- represents the index of the current element being processed in the array.

array

- represents the array itself, allowing access to the entire array within the callback function.

The

context

is the object that should be passed as the function context parameter to the

callback

function, ensuring that the

this

keyword within the

callback

function refers to this

context

object.

Try to implement it without using the built-in array methods.

Example 1:

Input:

arr = [1,2,3], callback = (val, i, arr) => arr[i] = val * 2, context = {"context":true}

Output:

[2,4,6]

Explanation:

arr.forEach(callback, context) console.log(arr) // [2,4,6]

The callback is executed on each element of the array.

Example 2:

Input:

arr = [true, true, false, false], callback = (val, i, arr) => arr[i] = this, context = {"context": false}

Output:

[{"context":false},{"context":false},{"context":false},{"context":false}]

Explanation:

arr.forEach(callback, context) console.log(arr) //
[{"context":false},{"context":false},{"context":false},{"context":false}]

The callback is executed on each element of the array with the right context.

Example 3:

Input:

arr = [true, true, false, false], callback = (val, i, arr) => arr[i] = !val, context = {"context": 5}

Output:

[false,false,true,true]

Constraints:

arr

is a valid JSON array

context

is a valid JSON object

fn

is a function

0 <= arr.length <= 10

5

Code Snippets

JavaScript:

```
/**
* @param {Function} callback
* @param {Object} context
* @return {void}
*/
Array.prototype.forEach = function(callback, context) {

}

/**
* const arr = [1,2,3];
* const callback = (val, i, arr) => arr[i] = val * 2;
* const context = {"context":true};
*
* arr.forEach(callback, context)
*
* console.log(arr) // [2,4,6]
*/
```

TypeScript:

```
type JSONValue = null | boolean | number | string | JSONValue[] | { [key:
string]: JSONValue };
type Callback = (currentValue: JSONValue, index: number, array: JSONValue[])
=> any
type Context = Record<string, JSONValue>

Array.prototype.forEach = function(callback: Callback, context: Context):
void {

}

/**
* const arr = [1,2,3];
* const callback = (val, i, arr) => arr[i] = val * 2;
* const context = {"context":true};
*
* arr.forEach(callback, context)
*
* console.log(arr) // [2,4,6]
*/
```

Solutions

JavaScript Solution:

```
/**
* Problem: Array Prototype ForEach
* 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 {Function} callback
* @param {Object} context
* @return {void}
*/
Array.prototype.forEach = function(callback, context) {
}

/**
* const arr = [1,2,3];
* const callback = (val, i, arr) => arr[i] = val * 2;
* const context = {"context":true};
*
* arr.forEach(callback, context)
*
* console.log(arr) // [2,4,6]
*/
```

TypeScript Solution:

```
/**
* Problem: Array Prototype ForEach
* 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
*/

type JSONValue = null | boolean | number | string | JSONValue[] | { [key:
string]: JSONValue };
type Callback = (currentValue: JSONValue, index: number, array: JSONValue[])
=> any
type Context = Record<string, JSONValue>

Array.prototype.forEach = function(callback: Callback, context: Context):
void {

}

/**
* const arr = [1,2,3];
* const callback = (val, i, arr) => arr[i] = val * 2;
* const context = {"context":true};
*
```

```
* arr.forEach(callback, context)
*
* console.log(arr) // [2,4,6]
*/
```