Problem 2805: Custom Interval

Problem Information

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

Problem Description

Function

customInterval

Given a function

fn

, a number

delay

and a number

period

, return a number

id

.

customInterval

is a function that should execute the provided function fn

at intervals based on a linear pattern defined by the formula

delay + period * count

.

The

count

in the formula represents the number of times the interval has been executed starting from an initial value of

0

.

Function

customClearInterval

Given the

id

.

id

is the returned value from the function

customInterval

.

customClearInterval

should stop executing provided function

fn

at intervals.

Note:

The

setTimeout

and

setInterval

functions in Node.js return an object, not a number.

Example 1:

Input:

delay = 50, period = 20, cancelTime = 225

Output:

[50,120,210]

Explanation:

const t = performance.now()   const result = []   const fn = () => {
result.push(Math.floor(performance.now() - t)) } const id = customInterval(fn, delay, period)
setTimeout(() => { customClearInterval(id) }, 225)

50 + 20 * 0 = 50 // 50ms - 1st function call 50 + 20 * 1 = 70 // 50ms + 70ms = 120ms - 2nd function call 50 + 20 * 2 = 90 // 50ms + 70ms + 90ms = 210ms - 3rd function call

Example 2:

Input:

delay = 20, period = 20, cancelTime = 150

Output:

[20,60,120]

Explanation:

20 + 20 * 0 = 20 // 20ms - 1st function call 20 + 20 * 1 = 40 // 20ms + 40ms = 60ms - 2nd function call 20 + 20 * 2 = 60 // 20ms + 40ms + 60ms = 120ms - 3rd function call

Example 3:

Input:

delay = 100, period = 200, cancelTime = 500

Output:

[100,400]

Explanation:

100 + 200 * 0 = 100 // 100ms - 1st function call 100 + 200 * 1 = 300 // 100ms + 300ms = 400ms - 2nd function call

Constraints:

20 <= delay, period <= 250

20 <= cancelTime <= 1000

Code Snippets

JavaScript:

```
/**
* @param {Function} fn
* @param {number} delay
* @param {number} period
* @return {number} id
*/

function customInterval(fn, delay, period){

}

/**
* @param {number} id
* @return {void}
*/
function customClearInterval(id) {

}
```

TypeScript:

```
function customInterval(fn: Function, delay: number, period: number): number
{

}

function customClearInterval(id: number): void {

}
```

Solutions

JavaScript Solution:

```
/**
* Problem: Custom Interval
* Difficulty: Medium
* Tags: math
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {Function} fn
* @param {number} delay
* @param {number} period
* @return {number} id
*/

function customInterval(fn, delay, period){

}

/**
* @param {number} id
* @return {void}
*/
function customClearInterval(id) {

}
```

TypeScript Solution:

```
/**
* Problem: Custom Interval
* Difficulty: Medium
* Tags: math
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

function customInterval(fn: Function, delay: number, period: number): number
{

}

function customClearInterval(id: number): void {


}
```