Problem 2796: Repeat String

Problem Information

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

Problem Description

Write code that enhances all strings such that you can call the

string.replicate(x)

method on any string and it will return repeated string

x

times.

Try to implement it without using the built-in method

string.repeat

.

Example 1:

Input:

str = "hello", times = 2

Output:

"hellohello"

Explanation:

"hello" is repeated 2 times

Example 2:

Input:

str = "code", times = 3

Output:

"codecodecode"

Explanation:

"code" is repeated 3 times

Example 3:

Input:

str = "js", times = 1

Output:

"js"

Explanation:

"js" is repeated 1 time

Constraints:

1 <= times <= 10

5

1 <= str.length <= 1000

Follow up:

Let's assume, for the sake of simplifying analysis, that concatenating strings is a constant time operation

O(1)

. With this assumption in mind, can you write an algorithm with a runtime complexity of

O(log n)

?

Code Snippets

JavaScript:

```
/**
* @param {number} times
* @return {string}
*/
String.prototype.replicate = function(times) {

}
```

TypeScript:

```
interface String {
replicate(times: number): string;
}

String.prototype.replicate = function(times): string {

}
```

Solutions

JavaScript Solution:

```
/**
* Problem: Repeat String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {number} times
* @return {string}
*/
String.prototype.replicate = function(times) {

}
```

TypeScript Solution:

```
/**
* Problem: Repeat String
* Difficulty: Easy
* Tags: string
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

interface String {
replicate(times: number): string;
}

String.prototype.replicate = function(times): string {

}
```