

303. Range Sum Query - Immutable

Description

Given an integer array `nums`, handle multiple queries of the following type:

1. Calculate the **sum** of the elements of `nums` between indices `left` and `right` **inclusive** where `left <= right`.

Implement the `NumArray` class:

- `NumArray(int[] nums)` Initializes the object with the integer array `nums`.
- `int sumRange(int left, int right)` Returns the **sum** of the elements of `nums` between indices `left` and `right` **inclusive** (i.e. `nums[left] + nums[left + 1] + ... + nums[right]`).

Example 1:

Input

```
["NumArray", "sumRange", "sumRange", "sumRange"]  
[[[-2, 0, 3, -5, 2, -1]], [0, 2], [2, 5], [0, 5]]
```

Output

```
[null, 1, -1, -3]
```

Explanation

```
NumArray numArray = new NumArray([-2, 0, 3, -5, 2, -1]);  
numArray.sumRange(0, 2); // return (-2) + 0 + 3 = 1  
numArray.sumRange(2, 5); // return 3 + (-5) + 2 + (-1) = -1  
numArray.sumRange(0, 5); // return (-2) + 0 + 3 + (-5) + 2 + (-1) = -3
```

Constraints:

- $1 \leq \text{nums.length} \leq 10^4$
- $-10^5 \leq \text{nums}[i] \leq 10^5$
- $0 \leq \text{left} \leq \text{right} < \text{nums.length}$
- At most 10^4 calls will be made to `sumRange`.

