303. Range Sum Query - Immutable

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 <= nums.length <= 104`
- `-105 <= nums[i] <= 105`
- `0 <= left <= right < nums.length`
- At most `104` calls will be made to `sumRange`.

```
def __init__(self, nums: List[int]):
        self.nums = nums
        self.dp = [0]*len(nums)
        self.dp[0] = self.nums[0]
        for i in range(1,len(self.nums)):
            self.dp[i] = self.dp[i-1]+self.nums[i]
```

```
    def sumRange(self, left: int, right: int) -> int:
        return self.dp[right]-self.dp[left]+self.nums[left]
```