**Problem #3010: Divide an array into subarrays with Minimum Cost I. (Easy)**

<https://leetcode.com/problems/divide-an-array-into-subarrays-with-minimum-cost-i/description/>

**My Solution:**

**Approach**

1. If the length of nums is 3, then return the sum of all the elements in nums.
2. Get nums at index 0. Sort nums from index 1 to end in ascending order and get the smallest two elements. Return the sum of these 3 numbers.

**Complexity**

* Time complexity: O(n \* log(n)) since we are sorting nums[1:]
* Space complexity: O(n)

**Code**

class Solution:

def minimumCost(self, nums: List[int]) -> int:

if len(nums) == 3:

return sum(nums)

return nums[0] + sum(sorted(nums[1:])[0:2])