Problem 3510: Minimum Pair Removal to Sort Array II

Problem Information

Difficulty: Hard
Acceptance Rate: 14.79%
Paid Only: No
Tags: Array, Hash Table, Linked List, Heap (Priority Queue), Simulation, Doubly-Linked List, Ordered Set

Problem Description

Given an array `nums`, you can perform the following operation any number of times:

* Select the **adjacent** pair with the **minimum** sum in `nums`. If multiple such pairs exist, choose the leftmost one. * Replace the pair with their sum.

Return the **minimum number of operations** needed to make the array **non- decreasing**.

An array is said to be **non-decreasing** if each element is greater than or equal to its previous element (if it exists).

Example 1:

Input: nums = [5,2,3,1]

Output: 2

Explanation:

* The pair `(3,1)` has the minimum sum of 4. After replacement, `nums = [5,2,4]`. * The pair `(2,4)` has the minimum sum of 6. After replacement, `nums = [5,6]`.

The array `nums` became non-decreasing in two operations.

Example 2:

Input: nums = [1,2,2]

Output: 0

Explanation:

The array `nums` is already sorted.

Constraints:

* `1 <= nums.length <= 105` * `-109 <= nums[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
int minimumPairRemoval(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int minimumPairRemoval(int[] nums) {

}
}
```

Python3:

```
class Solution:
def minimumPairRemoval(self, nums: List[int]) -> int:
```