Problem 3165: Maximum Sum of Subsequence With Non-adjacent Elements

Problem Information

Difficulty: Hard
Acceptance Rate: 15.21%
Paid Only: No
Tags: Array, Divide and Conquer, Dynamic Programming, Segment Tree

Problem Description

You are given an array `nums` consisting of integers. You are also given a 2D array `queries`, where `queries[i] = [posi, xi]`.

For query `i`, we first set `nums[posi]` equal to `xi`, then we calculate the answer to query `i` which is the **maximum** sum of a subsequence of `nums` where **no two adjacent elements are selected**.

Return the _sum_ of the answers to all queries.

Since the final answer may be very large, return it **modulo** `109 + 7`.

A **subsequence** is an array that can be derived from another array by deleting some or no elements without changing the order of the remaining elements.

Example 1:

Input: nums = [3,5,9], queries = [[1,-2],[0,-3]]

Output: 21

Explanation: After the 1st query, `nums = [3,-2,9]` and the maximum sum of a subsequence with non-adjacent elements is `3 + 9 = 12`. After the 2nd query, `nums = [-3,-2,9]` and the maximum sum of a subsequence with non-adjacent elements is 9.

Example 2:

Input: nums = [0,-1], queries = [[0,-5]]

Output: 0

Explanation: After the 1st query, `nums = [-5,-1]` and the maximum sum of a subsequence with non-adjacent elements is 0 (choosing an empty subsequence).

Constraints:

* `1 <= nums.length <= 5 * 104` * `-105 <= nums[i] <= 105` * `1 <= queries.length <= 5 * 104` * `queries[i] == [posi, xi]` * `0 <= posi <= nums.length - 1` * `-105 <= xi <= 105`

Code Snippets

C++:

```
class Solution {
public:
int maximumSumSubsequence(vector<int>& nums, vector<vector<int>>& queries) {

}
};
```

Java:

```
class Solution {
public int maximumSumSubsequence(int[] nums, int[][] queries) {

}
}
```

Python3:

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