907. Sum of Subarray Minimums

Given an array of integers arr, find the sum of min(b), where b ranges over every (contiguous) subarray of arr. Since the answer may be large, return the answer **modulo** $10^9 + 7$.

Example 1:

Input: arr = [3,1,2,4]

Output: 17

Explanation:

Subarrays are [3], [1], [2], [4], [3,1], [1,2], [2,4], [3,1,2], [1,2,4], [3,1,2,4].

Minimums are 3, 1, 2, 4, 1, 1, 2, 1, 1, 1.

Sum is 17.

Example 2:

Input: arr = [11,81,94,43,3]

Output: 444

Constraints:

- 1 <= arr.length <= $3 * 10^4$
- 1 <= arr[i] <= $3 * 10^4$

```
class Solution {
public:
    int sumSubarrayMins(vector<int>& arr) {
        int n = arr.size();
        vector<int> PLE(n);
        vector<int> NLE(n);
        stack<int> s;
        const int MOD = 1e9 + 7;

        for(int i=0;i<n;++i){
            while(!s.empty() && arr[s.top()] >= arr[i])
                s.pop();

            if(s.empty())
                PLE[i] = -1;
            else
                PLE[i] = s.top();
            s.push(i);
        }

        while(!s.empty())
            s.pop();

        for(int i=n-1;i>=0;--i){
            while(!s.empty() && arr[s.top()] > arr[i])
                s.pop();
            if(s.empty())
                NLE[i] = n;
            else
                NLE[i] = s.top();
            s.push(i);
        }

        long long sum = 0;
        for(int i=0;i<n;++i){
            long long left = i - PLE[i];
            long long right = NLE[i] - i;
            sum = (sum + arr[i] * left * right)%MOD;
        }

        return sum;
    }
};
```