Monotone stack

975. Odd Even Jump

Hard 737 212 Add to List Share

You are given an integer array `A`. From some starting index, you can make a series of jumps. The (1st, 3rd, 5th, ...) jumps in the series are called *odd numbered jumps*, and the (2nd, 4th, 6th, ...) jumps in the series are called *even numbered jumps*.

You may from index `i` jump forward to index `j` (with `i < j`) in the following way:

- During odd numbered jumps (ie. jumps 1, 3, 5, ...), you jump to the index `j` such that `A[i] <= A[j]` and `A[j]` is the smallest possible value. If there are multiple such indexes `j`, you can only jump to the **smallest** such index `j`.
- During even numbered jumps (ie. jumps 2, 4, 6, ...), you jump to the index `j` such that `A[i] >= A[j]` and `A[j]` is the largest possible value. If there are multiple such indexes `j`, you can only jump to the **smallest** such index `j`.
- (It may be the case that for some index `i`, there are no legal jumps.)

A starting index is *good* if, starting from that index, you can reach the end of the array (index `A.length - 1`) by jumping some number of times (possibly 0 or more than once.)

Return the number of good starting indexes.

Monotone stack

Odd: 다음에 위치하는 가장 가까우면서 크거나 같은 값

Even: 다음에 위치하는 가장 가까우면서 크거나 작은 값

위의 두 조건 때문에 이동할 수 있는 위치는 무조건 하나로 고정됨
DP 를 이용하면 뒤에서 부터 탐색하면서 끝까지 이동 가능한지 손쉽게 알 수 있음 .
DP[n][2] → n 번째 위치의 홀수 / 짝수 이동일 때 끝까지 이동 가능한가 ? => O(N)
그럼 과연 다음 이동할 수 있는 위치를 어떻게 알 수 있을까 ?

Monotone stack

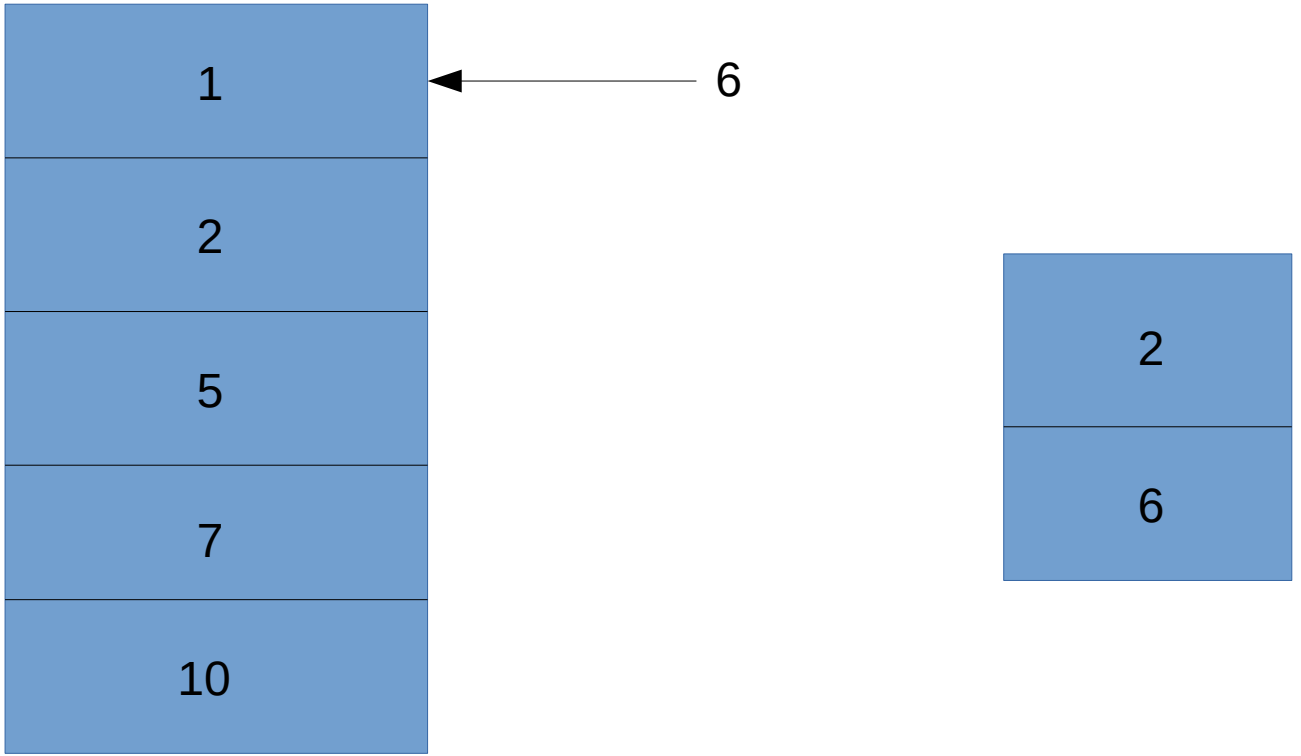
Naive solution: Brute force → O(n^2) => TLE

Use sort!: pair<value,index> 를 사용하여 정렬 후 알아낼 수 있다 . → O(NlogN)

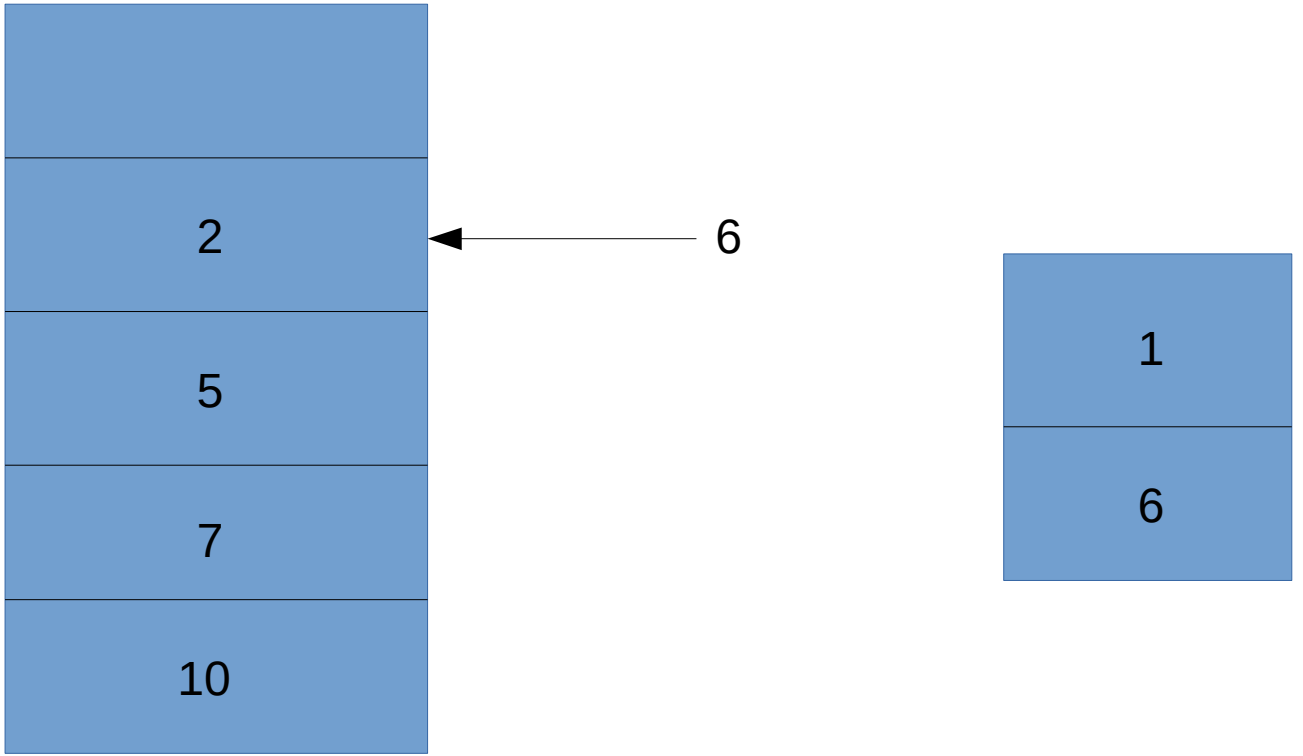
문제를 단순화 : 정렬된 배열 중 자신보다 크거나 같은 값 / 작거나 같은 값 탐색

Monotone stack: index 값이 감소하도록 저장

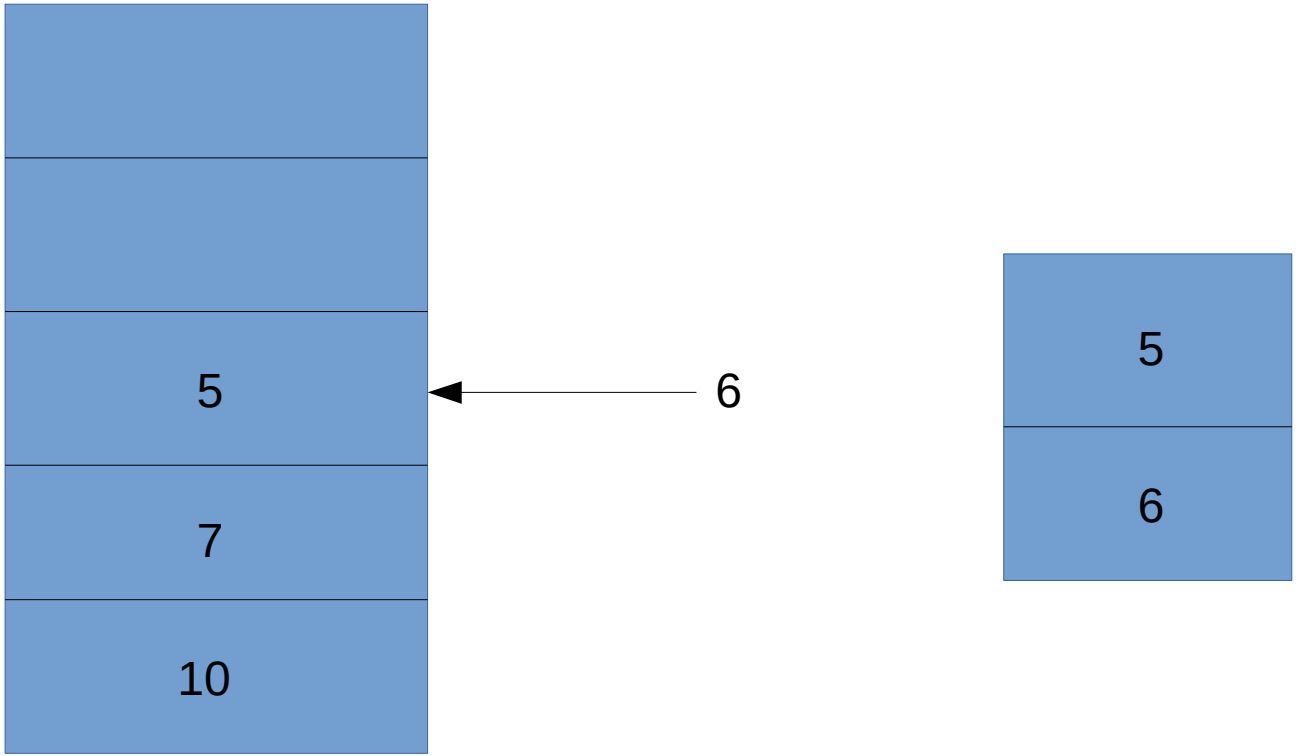
Monotone stack



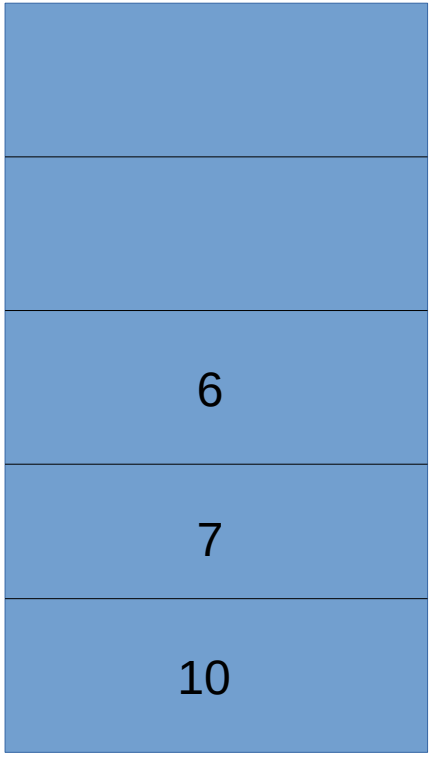
Monotone stack



Monotone stack



Monotone stack



Monotone stack

```
class Solution {
public:
    bool cmp(pair<int,int> &a, pair<int,int> &b){
        if(a.first==b.first)    return a.second<b.second;
        else    return a.first<b.first;
    }

    int oddEvenJumps(vector<int>& A) {
        int n=A.size();
        vector<int> odd(n,-1),even(n,-1);
        stack<int> nxt_odd,nxt_even;
        vector<vector<bool>> dp(n,vector<bool>(2));
        int ret=0;

        dp[n-1][0] = dp[n-1][1] = true;

        vector<pair<int,int>> B(n);
        for(int i=0;i<n;i++){
            B[i].first = A[i];
            B[i].second = i;
        }

        sort(B.begin(),B.end());
        for(int i=0;i<n;i++){
            while(!nxt_odd.empty() && nxt_odd.top() < B[i].second){
                int tmp = nxt_odd.top();
                odd[tmp] = B[i].second;
                nxt_odd.pop();
            }
            nxt_odd.push(B[i].second);
        }
        for(int i=0;i<n;i++)    B[i].first *= -1;
        sort(B.begin(),B.end());
```

```
        for(int i=0;i<n;i++){
            while(!nxt_even.empty() && nxt_even.top() < B[i].second){
                int tmp = nxt_even.top();
                even[tmp] = B[i].second;
                nxt_even.pop();
            }
            nxt_even.push(B[i].second);
        }

        for(int i=n-1;i>=0;i--){
            if(odd[i]!=-1 && dp[odd[i]][1]) dp[i][0] = true;
            if(even[i]!=-1 && dp[even[i]][0])   dp[i][1] = true;
        }
        for(int i=0;i<n;i++){
            if(dp[i][0])    ret++;
        }

        return ret;
    }
```

Monotone stack

907. Sum of Subarray Minimums

Medium 1357 84 Add to List Share

Given an array of integers `A`, find the sum of `min(B)`, where `B` ranges over every (contiguous) subarray of `A`.

Since the answer may be large, **return the answer modulo `10^9 + 7`.**

Example 1:

```
Input: [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.
```

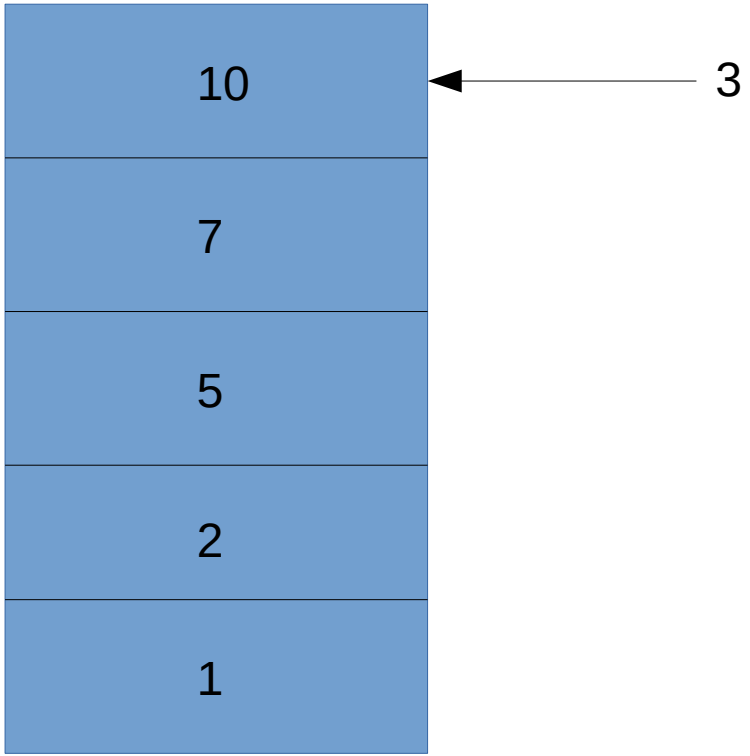
Monotone stack

Naive solution: Brute force → O(n^2) => TLE

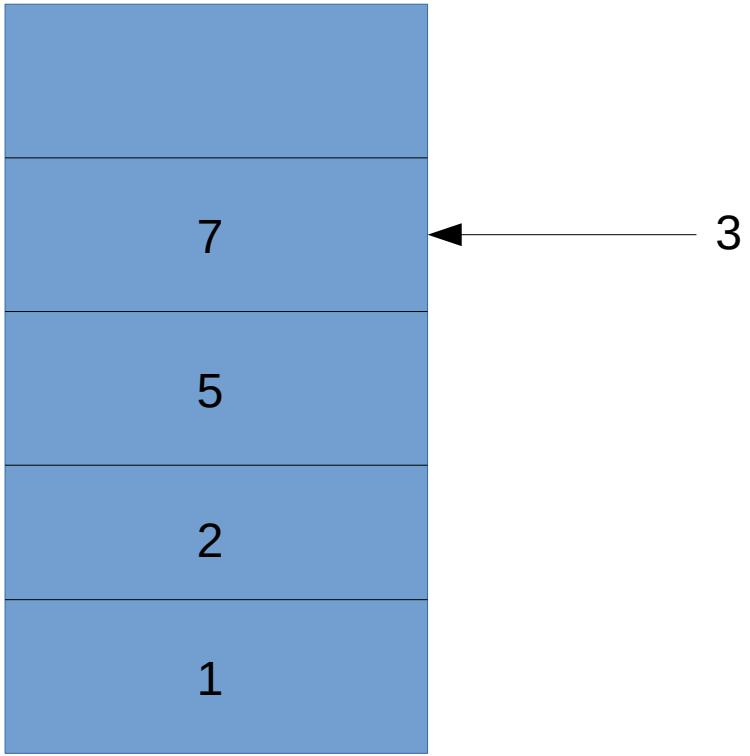
현재 값이 가장 작을 수 있는 범위를 구해야 함

How?: stack 2 개 이용 !

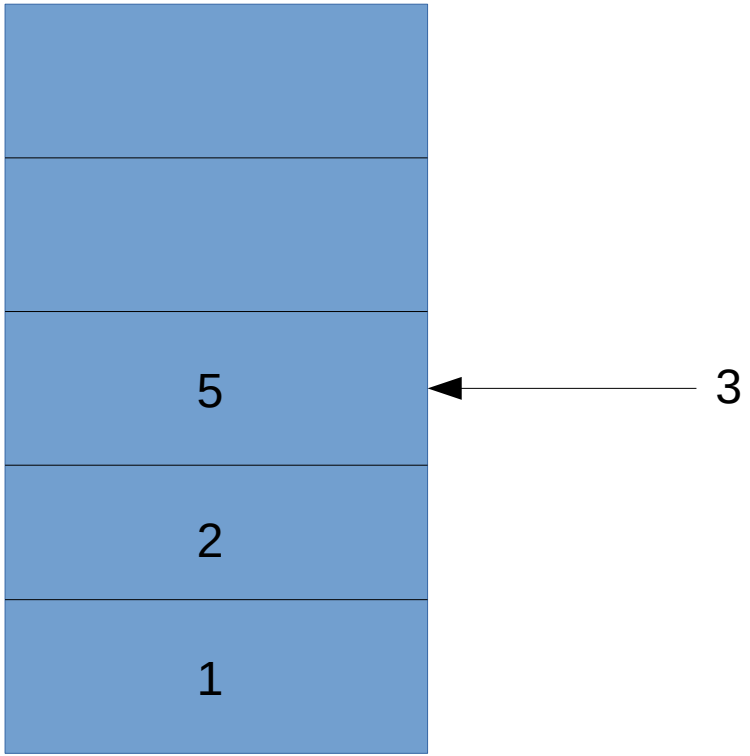
Monotone stack



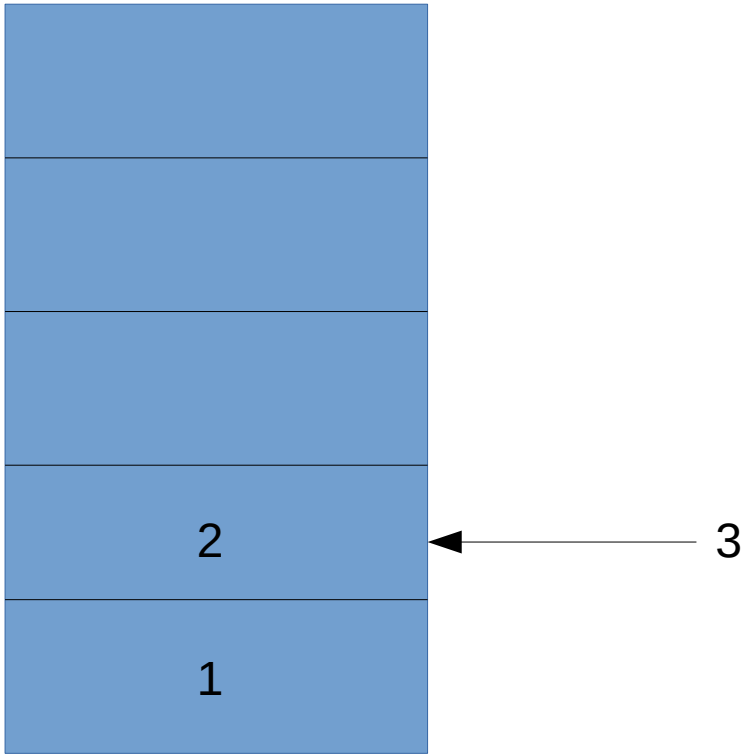
Monotone stack



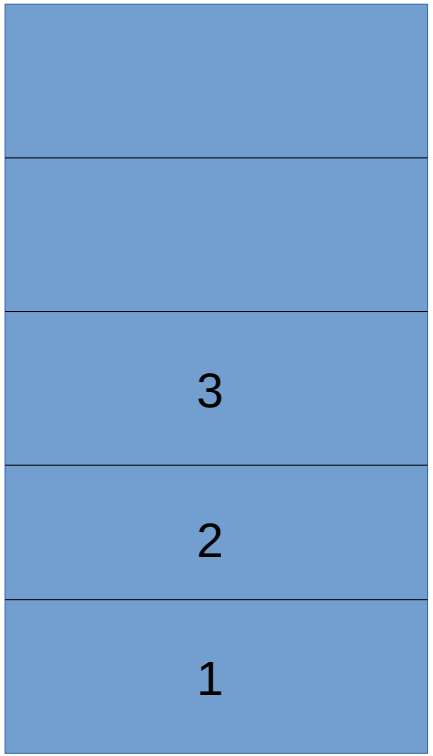
Monotone stack

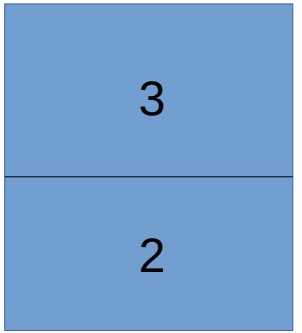


Monotone stack



Monotone stack





Monotone stack

```cpp
#include <bits/stdc++.h>

using namespace std;

class Solution {
public:
    int sumSubarrayMins(vector<int>& A) {
        int ret=0,n=A.size();
        vector<int> left(n), right(n);
        stack<int> prev,nxt;

        for(int i=0;i<n;i++)    right[i] = n-i;

        for(int i=0;i<n;i++){
            while(!prev.empty() && A[prev.top()] > A[i]) prev.pop();
            left[i] = prev.empty() ? i+1 : i-prev.top();
            prev.push(i);

            while(!nxt.empty() && A[nxt.top()] > A[i]){
                int tmp = nxt.top();
                right[tmp] = i-tmp;
                nxt.pop();
            }
            nxt.push(i);
        }
        for(int i=0;i<n;i++)    ret = (ret+A[i]*left[i]*right[i])%1000000007;
        return ret;
    }
};
```