Official

F - Sums of Sliding Window Maximum (/contests/abc407/tasks/abc407_f) Editorial by **en_translator** (/users/en_translator)

Let us define the ordering of the elements of A by the lexicographical order of (A_i, i). This way, the ordering of any pair of elements is uniquely determined.

For a (contiguous) subarray whose maximum value is A_i, let l be the number of elements to the left of A_i in the subarray, and r to the right. What condition should (l, r) satisfy?

Let L_i be the number of consecutive elements less than A_i to the left of A_i in A, and define R_i to the right likewise. (We will describe how to find these values afterward.)

Then, the subarray corresponding to (l, r) has A_i as the maximum value if and only if $0 \le l \le L_i$ and $0 \le r \le R_i$.

Here, write $x_{min} := \min(l, r)$ and $x_{max} := \max(l, r)$. Then the contribution of A_i to ans can be written as follows:

- For $0 < k \le 1 + x_{min}$, add $k \times A_i$ to $ans[k]$.
- For $1 + x_{min} < k \le 1 + x_{max}$, add $(1 + x_{min}) \times A_i$ to $ans[k]$.
- For $1 + x_{max} < k \le 2 + x_{min} + x_{max}$, add $(2 + x_{min} + x_{max} - k) \times A_i$ to $ans[k]$.

These are all affine function with a constant slope.

The cumulative sum trick is usually used to add a value to a specific subarray of a sequence, but it can be extended to affine functions as follows.

Let us replace a range addition as follows:

- Add $+A_i$ to $ans^{(2)}[1]$.
- Add $-A_i$ to $ans^{(2)}[2 + x_{min}]$.
- Add $-A_i$ to $ans^{(2)}[2 + x_{max}]$.
- Add $+A_i$ to $ans^{(2)}[2 + x_{min} + x_{max}]$.

After performing these operation for all ranges, repeat computing the cumulative sums twice, as follows:

- Let $ans^{(1)}[0] = ans^{(2)}[0]$ and $ans^{(1)}[i+1] := ans^{(1)}[i] + ans^{(2)}[i+1]$.
- $ans[0] = ans^{(1)}[0],\ ans[i+1] := ans[i] + ans^{(1)}[i+1]$.

This way, we can find ans.

How to find L_i and R_i?

Initialize the set of indices S by $\{0, N+1\}$.

Inspect A_i in descending order of the value, and perform the following operation for each element:

1. Let L' be the maximum value less than i in S, and R' be the minimum value greater than i.
2. Let $L_i = i - L' - 1$ and $R_i = R' - i - 1$.
3. Insert i to S.

To perform step 1., we can use `std::set` in the standard library of C++, utilizing the `set.lower_bound` function.

The following is sample code in C++.

Copy

2025-05-24 (Sat)
23:47:07 -04:00

```cpp
#include <iostream>
#include <string>
#include <vector>
#include <set>
#include <array>
using std::cin;
using std::cout;
using std::cerr;
using std::endl;
using std::string;
using std::pair;
using std::vector;
using std::set;
using std::min;
using std::max;
using std::array;
#include <atcoder/all>


typedef long long ll;
typedef pair<ll, ll> P;

vector<ll> solve (const ll n, const vector<ll> &a) {
        vector<ll> ans(n+1, 0);

        vector<P> asort(n);
        for (ll i = 0; i < n; i++) {
                asort[i] = {a[i], i};
        }
        sort(asort.begin(), asort.end());

        array<vector<ll>, 3> integral;

        integral[0].resize(n+3);
        for (ll i = 0; i <= n+2; i++) {
                integral[0][i] = 0;
        }
        set<ll> used = {-1, n};
        for (ll i = n-1; i >= 0; i--) {
                ll val = asort[i].first;
                ll idx = asort[i].second;

                auto itgeq = used.lower_bound(idx);
                auto itle = itgeq; --itle;

                // how many elements (on left/right) can a[idx] absorb as a maximum
```

2025-05-24 (Sat)
23:47:07 -04:00

```
                ll r = *itgeq - idx - 1;
                ll l = idx - *itle - 1;

                ll xmin = min(l, r), xmax = max(l, r);

                // ans[0 < i <= 1+min] <- i
                // ans[1+min < i <= 1+max] <- 1+min
                // ans[1+max < i <= 1+min+max] <- 1+min - (i - (1+max))
                // ans[1+min+max < i] <- 0

                // ans'[0 < i <= 1+min] <- +1
                // ans'[1+min < i <= 1+max] <- 0
                // ans'[1+max < i <= 1+min+max] <- -1
                // ans'[1+min+max < i] <- 0

                // ans''[1] <- +1
                // ans''[1+min+1] <- -1
                // ans''[1+max+1] <- -1
                // ans''[1+min+max+1] <- +1
                integral[0][1            ] += val * (+1);
                integral[0][1+xmin+1     ] += val * (-1);
                integral[0][1+xmax+1     ] += val * (-1);
                integral[0][1+xmin+xmax+2] += val * (+1);

                used.insert(idx);
        }
        for (ll order = 1; order <= 2; order++) {
                integral[order].resize(n+3);
                integral[order][0] = 0;
                for (ll i = 1; i <= n+2; i++) {
                        integral[order][i] = integral[order][i-1] + integral[order-1][i];
                }
        }

        for (ll i = 1; i <= n; i++) {
                ans[i] = integral[2][i];
        }


        return ans;
}


int main (void) {
        ll n;
        cin >> n;
```

```
        vector<ll> a(n);
        for (ll i = 0; i < n; i++) {
                cin >> a[i];
        }
        vector<ll> anslist = solve(n, a);
        for (ll i = 1; i <= n; i++) {
                cout << anslist[i] << "\n";
        }


        return 0;
}
```

posted: 44 minutes ago
last update: 44 minutes ago

am)

ps%3A%2F%2Fatcoder.jp%2Fcontests%2Fabc407%2Feditorial%2F13112%3Flang%3Den&title=Editorial%20-
)407)

2025-05-24 (Sat)
23:47:07 -04:00