**Inversion - 2**

**Problem**

We are given a permutations pi of n elements. For each i, a[i] denotes the number of j such that

*j < i*

*a[j] > a[i]*

Our task is to restore the original permutation.

Example input

5

0 1 1 0 3

Output

4 1 3 5 2

**Idea and Brainstorming**

In our “present sir” approach, at the end all the numbers in the number line will be updated to 1. We have to build our answer in the reverse order.

**Approach**

1. Iterating the given permutation array in reverse order.
2. We need to find the index of the kth element from the end and mark it as 0 on the number line.
3. Keep adding the responses to the answers array and in the end reverse the answer array.

**Code**

#include "bits/stdc++.h"

using namespace std;

#define int long long

const int N = 1e5+2, MOD = 1e9+7;

int tree[4\*N], a[N];

void build(int node, int st, int en)

{

if(st == en){

tree[node] = a[st];

return;

}

int mid = (st + en)/2;

build(2\*node, st, mid);

build(2\*node+1, mid+1, en);

tree[node] = tree[2\*node] + tree[2\*node+1];

}

int query(int node, int st, int en, int k){

if(st == en)

return st;

int mid = (st+en)/2;

if(k<tree[2\*node]){

return query(2\*node, st,mid, k);

}

else

{

return query(2\*node+1, mid+1, en, k-tree[2\*node]);

}

}

void update(int node, int st, int en, int idx, int val){

if(st == en){

a[st] = val;

tree[node] = val;

return;

}

int mid = (st+en)/2;

if(idx <= mid){

update(2\*node, st, mid, idx, val);

}

else

{

update(2\*node+1, mid+1, en, idx, val);

}

tree[node] = tree[2\*node] + tree[2\*node+1];

}

signed main()

{

int n;

cin >> n;

for(int i=0; i<n; i++){

a[i] = 1;

}

build(1,0,n-1);

vector<int> b(n);

for(int i=0; i<n; i++){

cin >> b[i];

}

int currPresentSirs = n;

vector<int> ans;

for(int i=n-1; i>=0; i--){

int k = currPresentSirs - b[i] - 1;

currPresentSirs--;

int temp = query(1,0,n-1,k);

update(1,0,n-1,temp, 0);

ans.push\_back(temp+1);

}

reverse(ans.begin(), ans.end());

for(int i=0; i<ans.size(); i++)

cout << ans[i] <<" ";

return 0;

}