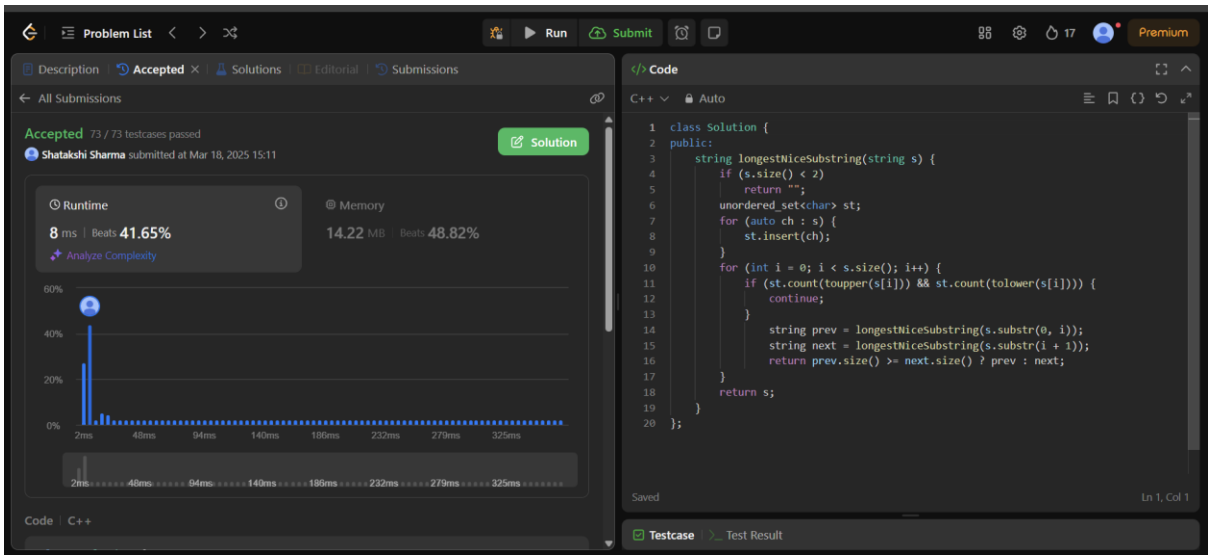
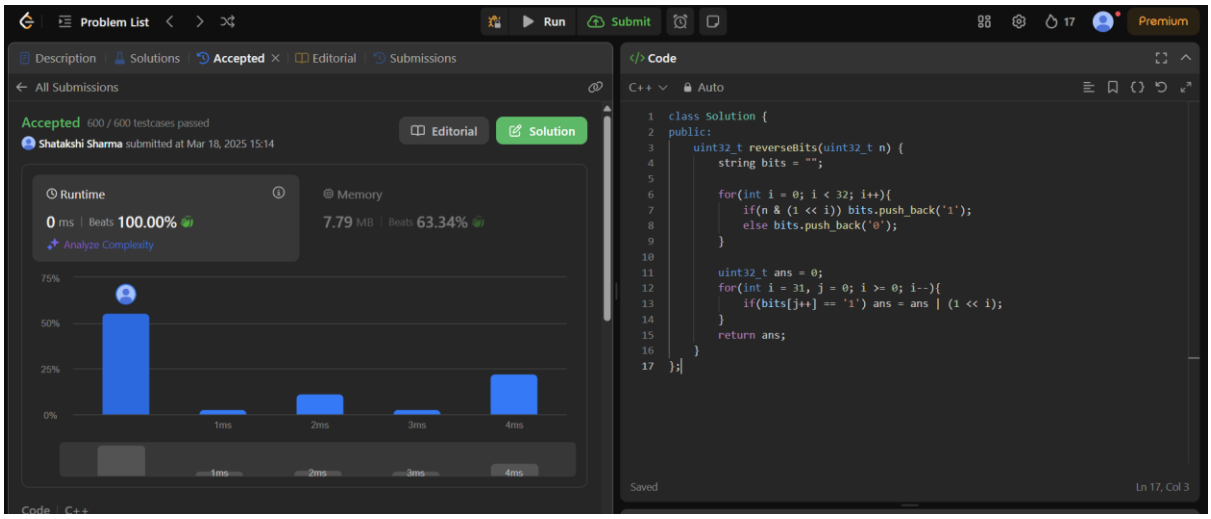
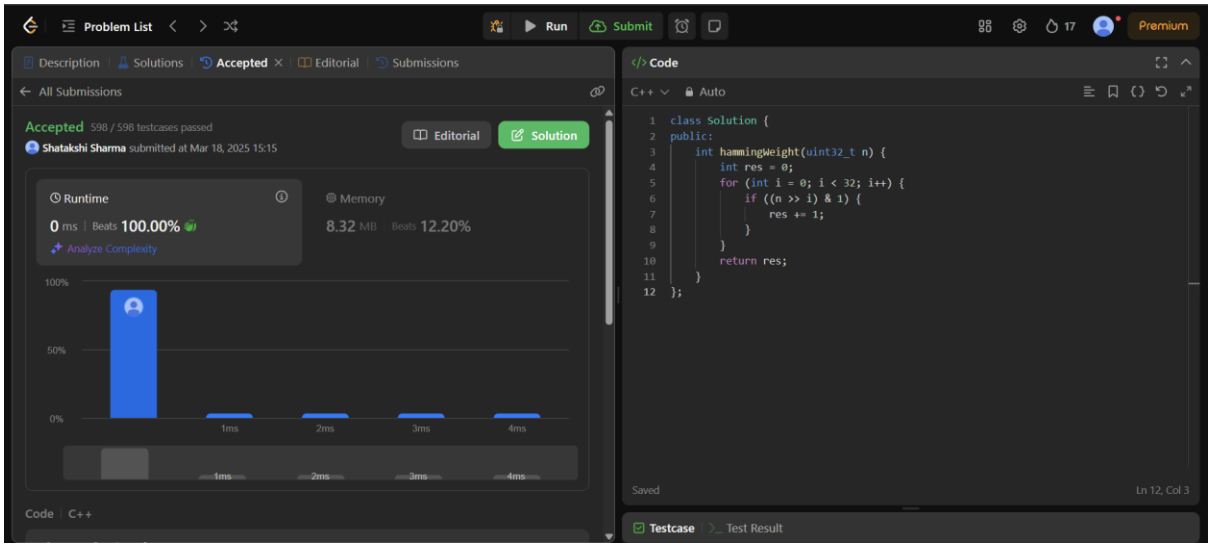
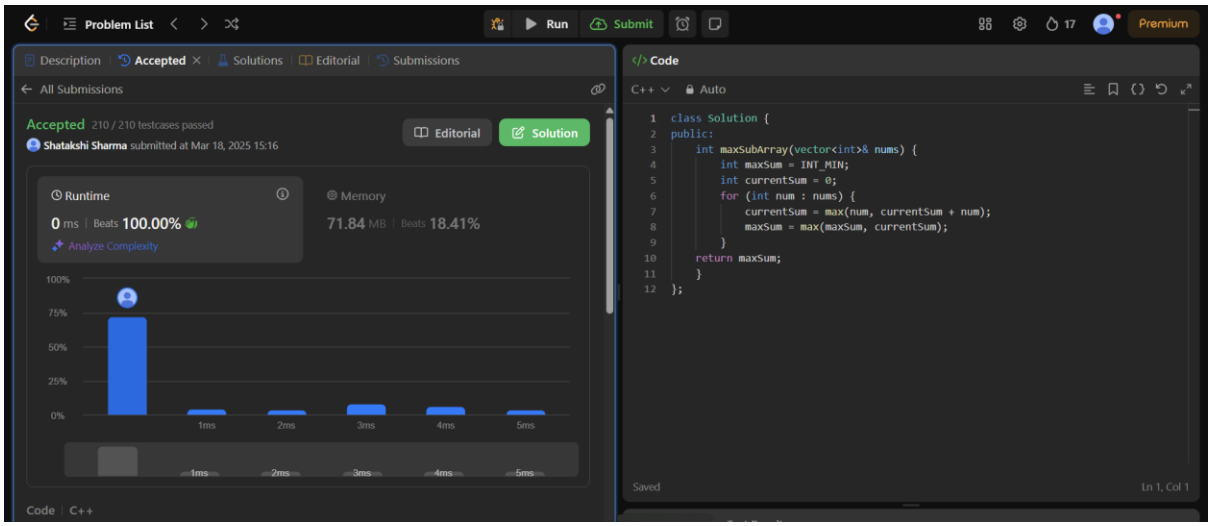
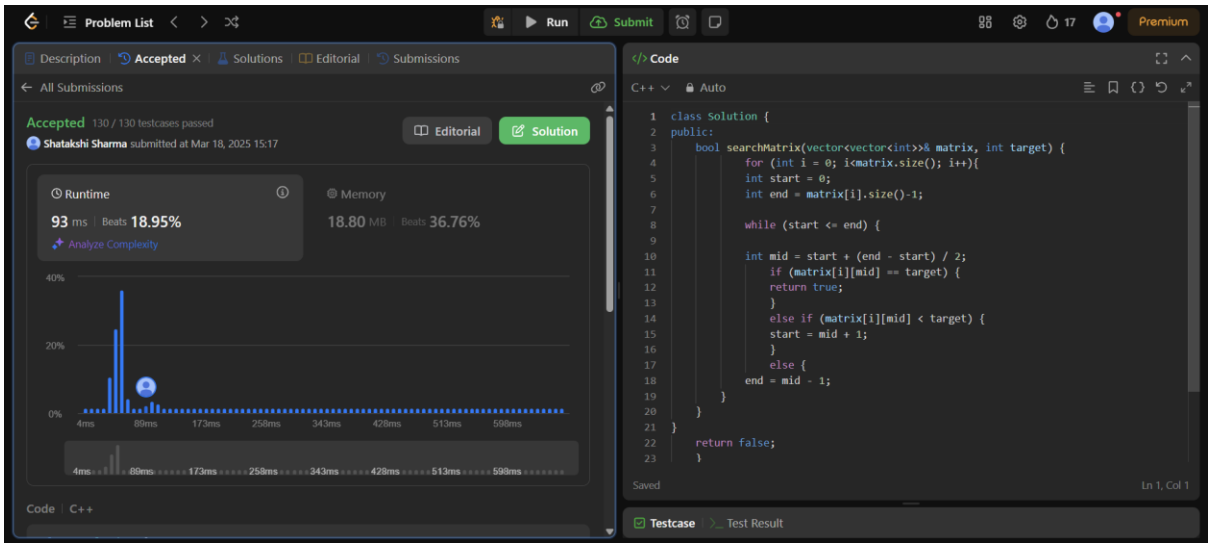
# ASSIGNMENT 4

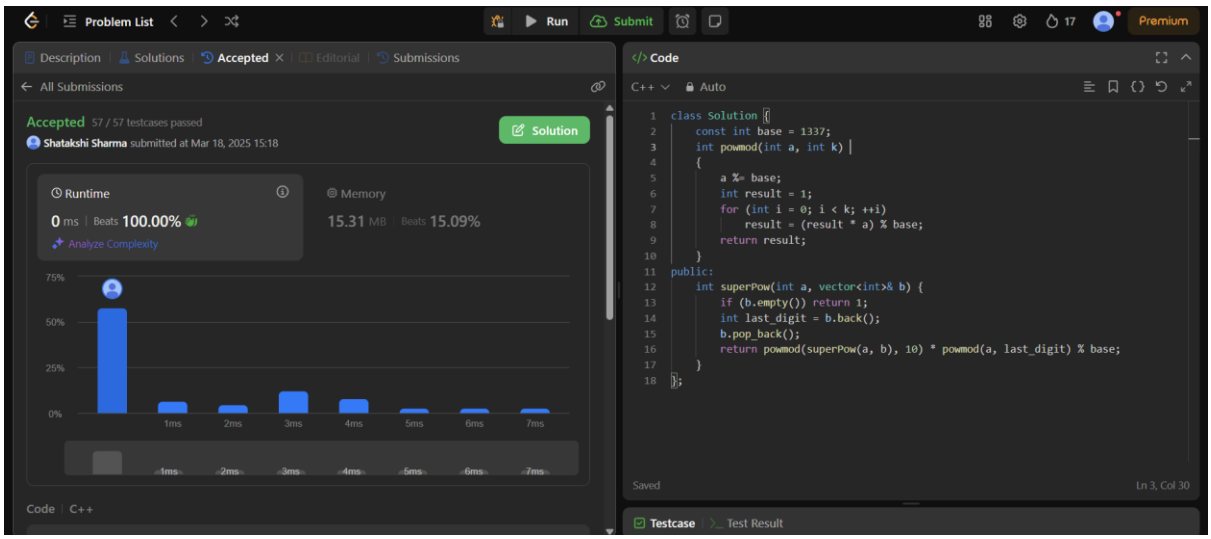


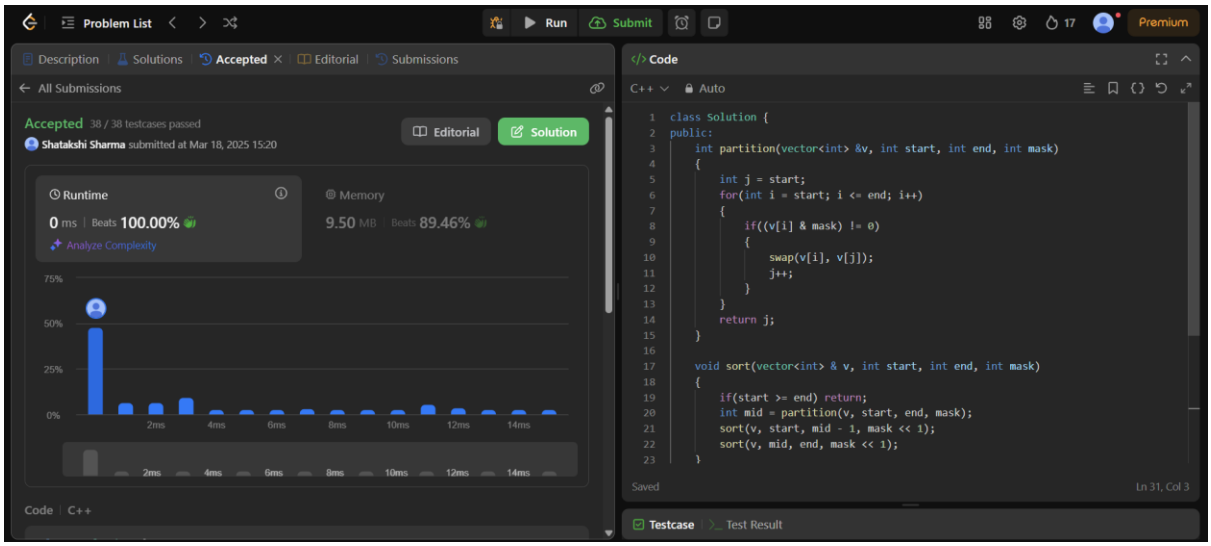



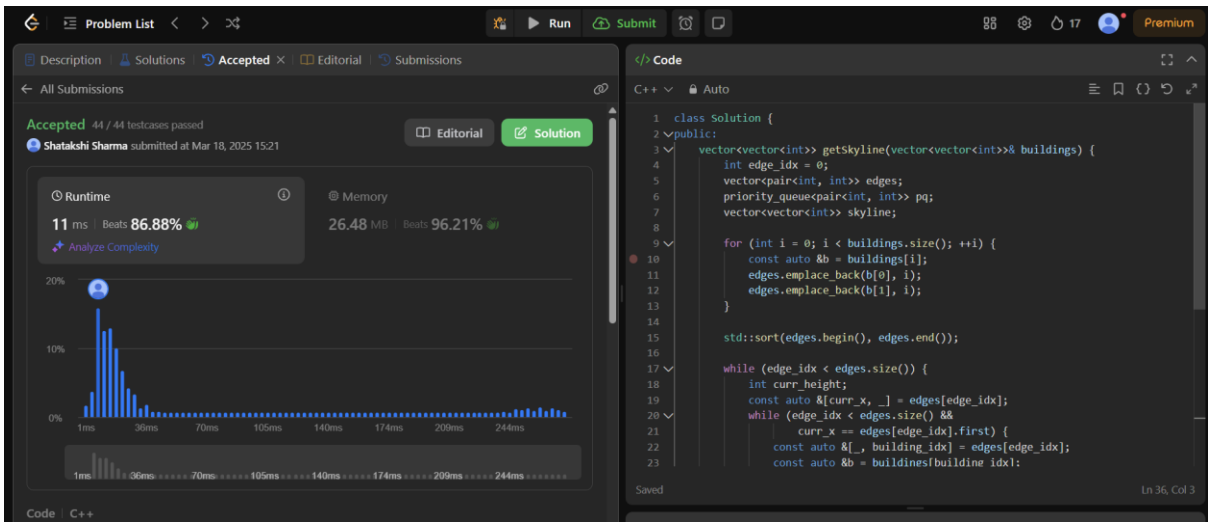



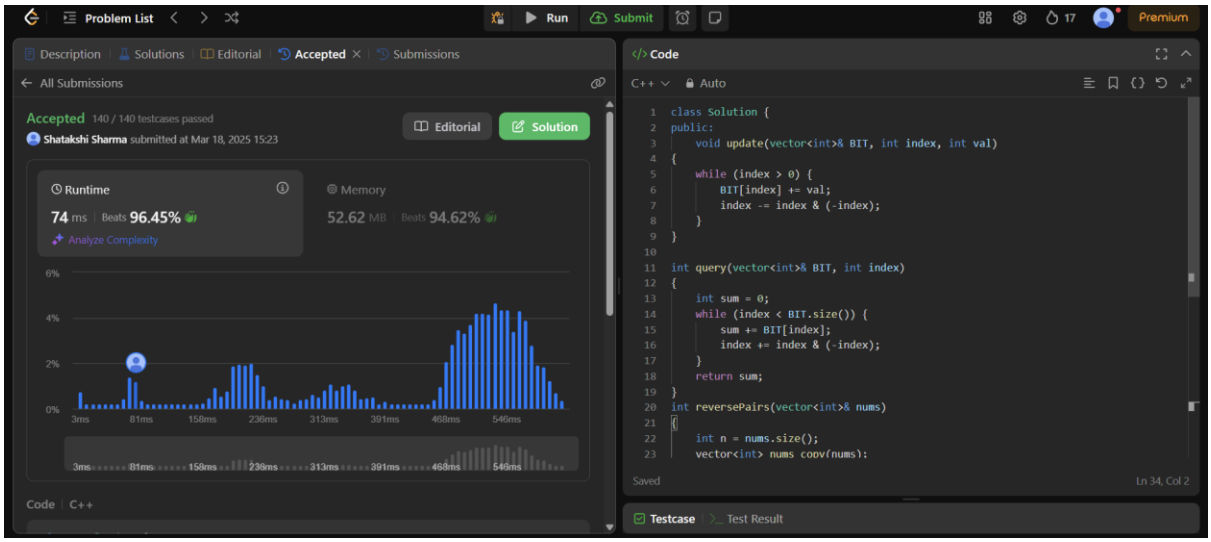

Problem List
Run
Submit
Premium
Description
Accepted
Solutions
Editorial
Submissions
All Submissions
Accepted 210 / 210 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:16
Editorial
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
71.84 MB | Beats 18.41%
Code
C++
Auto
class Solution {
public:
int maxSubArray(vector<int>& nums) {
int maxSum = INT_MIN;
int currentSum = 0;
for (int num : nums) {
currentSum = max(num, currentSum + num);
maxSum = max(maxSum, currentSum);
}
return maxSum;
}
};
Saved
Ln 1, Col 1

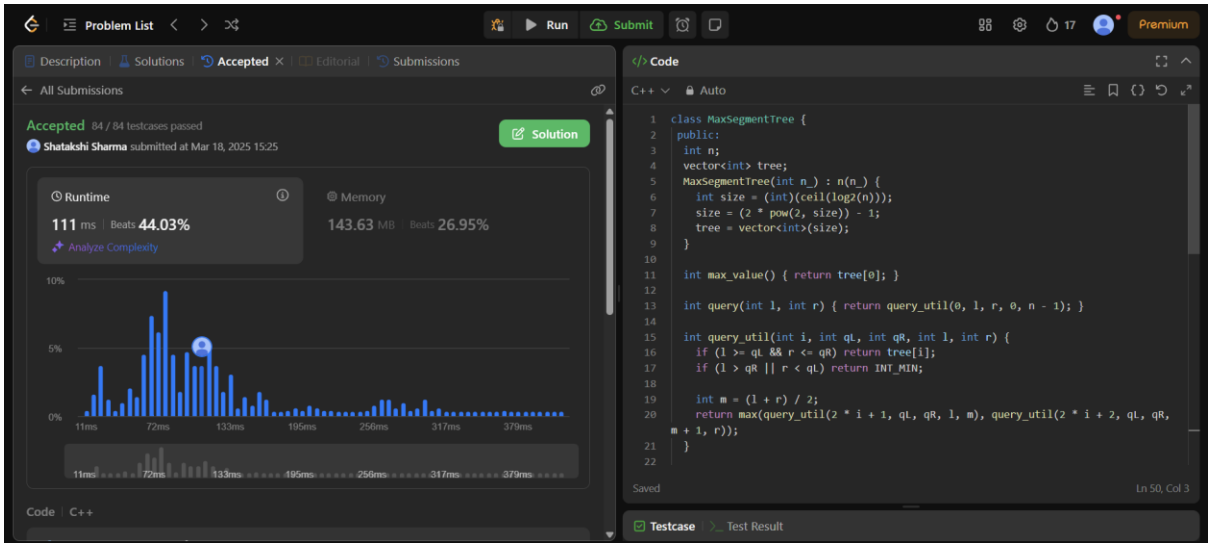

Problem List
Run
Submit
Premium
Description
Accepted
Solutions
Editorial
Submissions
All Submissions
Accepted 130 / 130 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:17
Editorial
Solution
Runtime
93 ms | Beats 18.95%
Analyze Complexity
Memory
18.80 MB | Beats 36.76%
Code
C++
Auto
class Solution {
public:
bool searchMatrix(vector<vector<int>>& matrix, int target) {
for (int i = 0; i<matrix.size(); i++){
int start = 0;
int end = matrix[i].size()-1;
while (start <= end) {
int mid = start + (end - start) / 2;
if (matrix[i][mid] == target) {
return true;
}
else if (matrix[i][mid] < target) {
start = mid + 1;
}
else {
end = mid - 1;
}
}
}
return false;
}
Saved
Ln 1, Col 1
Testcase
Test Result


Problem List
Run
Submit
Premium
Description
Solutions
Accepted
Editorial
Submissions
All Submissions
Accepted 57 / 57 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:18
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
15.31 MB | Beats 15.09%
Code
C++
Auto
class Solution {
const int base = 1337;
int powmod(int a, int k)
{
a %= base;
int result = 1;
for (int i = 0; i < k; ++i)
result = (result * a) % base;
return result;
}
public:
int superPow(int a, vector<int>& b) {
if (b.empty()) return 1;
int last_digit = b.back();
b.pop_back();
return powmod(superPow(a, b), 10) * powmod(a, last_digit) % base;
}
};
Saved
Ln 3, Col 30
Testcase
Test Result


Problem List
Run
Submit
Premium
Description
Solutions
Accepted
Editorial
Submissions
All Submissions
Accepted 38 / 38 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:20
Editorial
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
9.50 MB | Beats 89.46%
Code
C++
Auto
class Solution {
public:
int partition(vector<int> &v, int start, int end, int mask)
{
int j = start;
for(int i = start; i <= end; i++)
{
if((v[i] & mask) != 0)
{
swap(v[i], v[j]);
j++;
}
}
return j;
}
void sort(vector<int> & v, int start, int end, int mask)
{
if(start >= end) return;
int mid = partition(v, start, end, mask);
sort(v, start, mid - 1, mask << 1);
sort(v, mid, end, mask << 1);
}
Saved
Ln 31, Col 3
Testcase
Test Result


Problem List
Run
Submit
Premium
Description
Solutions
Accepted
Editorial
Submissions
All Submissions
Accepted 44 / 44 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:21
Editorial
Solution
Runtime
11 ms | Beats 86.88%
Analyze Complexity
Memory
26.48 MB | Beats 96.21%
Code
C++
Auto
class Solution {
public:
vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
int edge_idx = 0;
vector<pair<int, int>> edges;
priority_queue<pair<int, int>> pq;
vector<vector<int>> skyline;
for (int i = 0; i < buildings.size(); ++i) {
const auto &b = buildings[i];
edges.emplace_back(b[0], i);
edges.emplace_back(b[1], i);
}
std::sort(edges.begin(), edges.end());
while (edge_idx < edges.size()) {
int curr_height;
const auto &[curr_x, _] = edges[edge_idx];
while (edge_idx < edges.size() &&
curr_x == edges[edge_idx].first) {
const auto &[_, building_idx] = edges[edge_idx];
const auto &b = buildings[building_idx];
Saved
Ln 36, Col 3


Problem List
Run
Submit
Premium
Description
Solutions
Editorial
Accepted
Submissions
All Submissions
Accepted 140 / 140 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:23
Editorial
Solution
Runtime
74 ms | Beats 96.45%
Analyze Complexity
Memory
52.62 MB | Beats 94.62%
Code
C++
Auto
class Solution {
public:
void update(vector<int>& BIT, int index, int val)
{
while (index > 0) {
BIT[index] += val;
index -= index & (-index);
}
}
int query(vector<int>& BIT, int index)
{
int sum = 0;
while (index < BIT.size()) {
sum += BIT[index];
index += index & (-index);
}
return sum;
}
int reversePairs(vector<int>& nums)
{
int n = nums.size();
vector<int> nums_copy(nums);
Saved
Ln 34, Col 2
Testcase
Test Result


Problem List
Run
Submit
17
Premium
Description
Solutions
Accepted
Editorial
Submissions
All Submissions
Accepted 84 / 84 testcases passed
Shatakshi Sharma submitted at Mar 18, 2025 15:25
Solution
Runtime
111 ms | Beats 44.03%
Analyze Complexity
Memory
143.63 MB | Beats 26.95%
10%
5%
0%
11ms
72ms
133ms
195ms
256ms
317ms
379ms
Code | C++
Code
C++
Auto
1 class MaxSegmentTree {
2 public:
3 int n;
4 vector<int> tree;
5 MaxSegmentTree(int n_) : n(n_) {
6 int size = (int)(ceil(log2(n)));
7 size = (2 * pow(2, size)) - 1;
8 tree = vector<int>(size);
9 }
10
11 int max_value() { return tree[0]; }
12
13 int query(int l, int r) { return query_util(0, l, r, 0, n - 1); }
14
15 int query_util(int i, int ql, int qR, int l, int r) {
16 if (l >= ql && r <= qR) return tree[i];
17 if (l > qR || r < qL) return INT_MIN;
18
19 int m = (l + r) / 2;
20 return max(query_util(2 * i + 1, ql, qR, l, m), query_util(2 * i + 2, ql, qR,
m + 1, r));
21 }
22
Saved
Ln 50, Col 3
Testcase
Test Result