Name: Anshul

UID: 22BCS16477

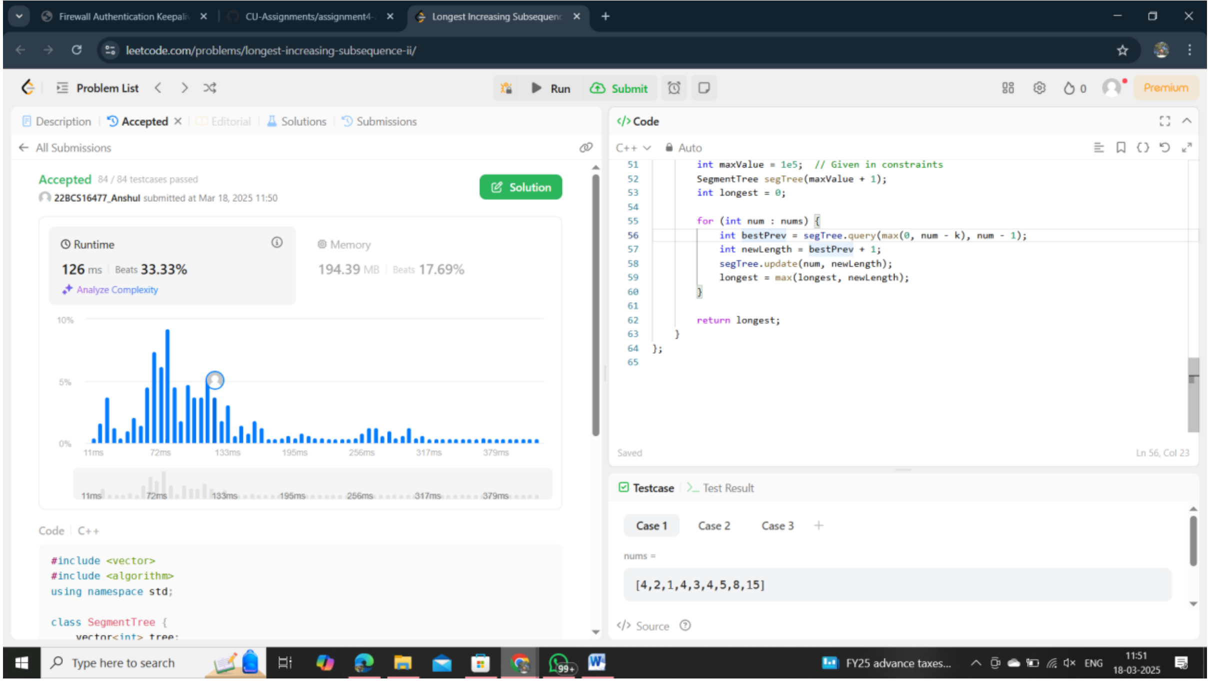
Section/Group: 609(B)

Longest Increasing Subsequence II

Code:

```
#include <vector>
#include <algorithm>
using namespace std;

class SegmentTree {
    vector<int> tree;
    int size;

public:
    SegmentTree(int n) {
        size = n;
        tree.resize(4 * n, 0);
    }

    // Query max in range [L, R]
    int query(int node, int start, int end, int L, int R) {
        if (R < start || end < L) return 0; // Out of range
        if (L <= start && end <= R) return tree[node]; // Fully in range
        int mid = (start + end) / 2;
        return max(query(2 * node, start, mid, L, R),
                   query(2 * node + 1, mid + 1, end, L, R));
    }

    // Update the segment tree at index idx
    void update(int node, int start, int end, int idx, int value) {
        if (start == end) {
            tree[node] = value;
        } else {
            int mid = (start + end) / 2;
            if (idx <= mid)
                update(2 * node, start, mid, idx, value);
            else
                update(2 * node + 1, mid + 1, end, idx, value);
            tree[node] = max(tree[2 * node], tree[2 * node + 1]);
        }
    }

    void update(int idx, int value) {
        update(1, 0, size - 1, idx, value);
    }
    int query(int L, int R) {
        if (L > R) return 0;
        return query(1, 0, size - 1, L, R);
    }
};

class Solution {
public:
    int lengthOfLIS(vector<int>& nums, int k) {
        int maxValue = 1e5;  // Given in constraints
        SegmentTree segTree(maxValue + 1);
        int longest = 0;

        for (int num : nums) {
            int bestPrev = segTree.query(max(0, num - k), num - 1);
            int newLength = bestPrev + 1;
            segTree.update(num, newLength);
            longest = max(longest, newLength);
        }

        return longest;
    }
};
```

Output:



Accepted 84 / 84 testcases passed

22BCS16477_Anshul submitted at Mar 18, 2025 11:50

Runtime 126 ms | Beats 33.33%

Memory 194.39 MB | Beats 17.69%

Code | C++

```
#include <vector>
#include <algorithm>
using namespace std;

class SegmentTree {
    vector<int> tree;
```

```
        int maxValue = 1e5;  // Given in constraints
        SegmentTree segTree(maxValue + 1);
        int longest = 0;

        for (int num : nums) {
            int bestPrev = segTree.query(max(0, num - k), num - 1);
            int newLength = bestPrev + 1;
            segTree.update(num, newLength);
            longest = max(longest, newLength);
        }

        return longest;
    }
};
```

Saved

Ln 56, Col 23

nums =

[4,2,1,4,3,4,5,8,15]

11:51
18-03-2025