Name: Atul

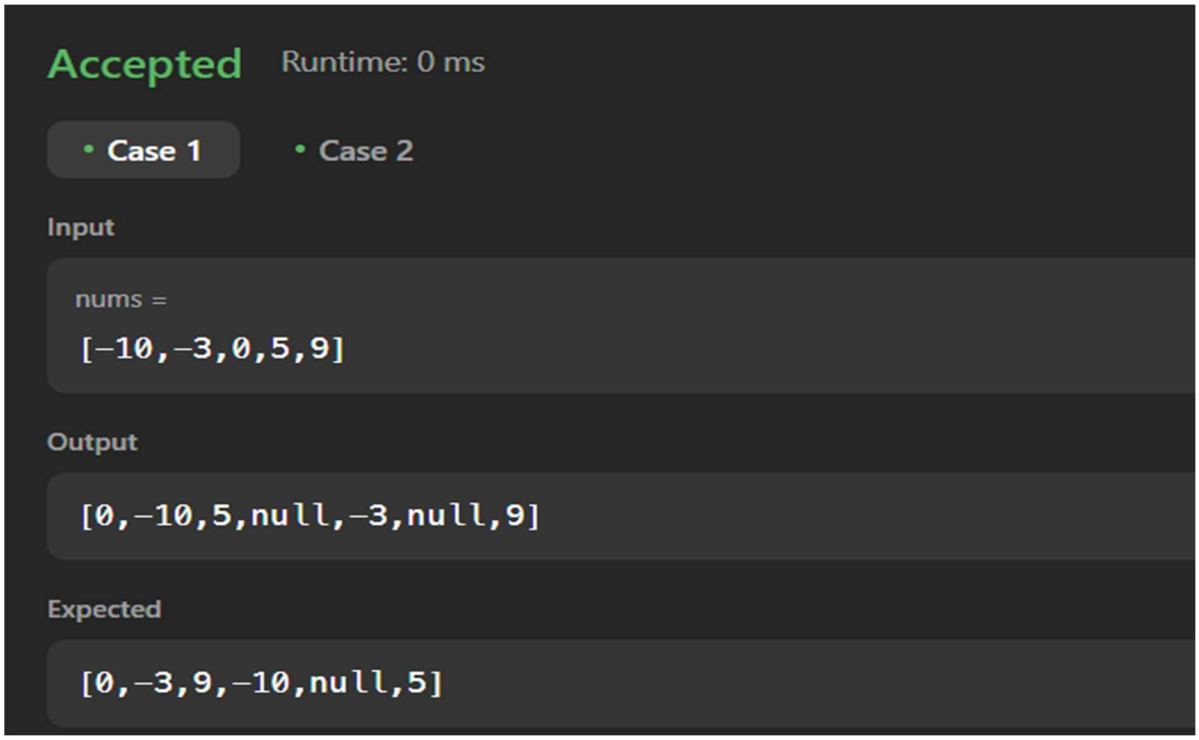
Uid:22BCS15834

Section:605-B

Q1. Convert Sorted Array to Binary Search Tree (LeetCode 108)

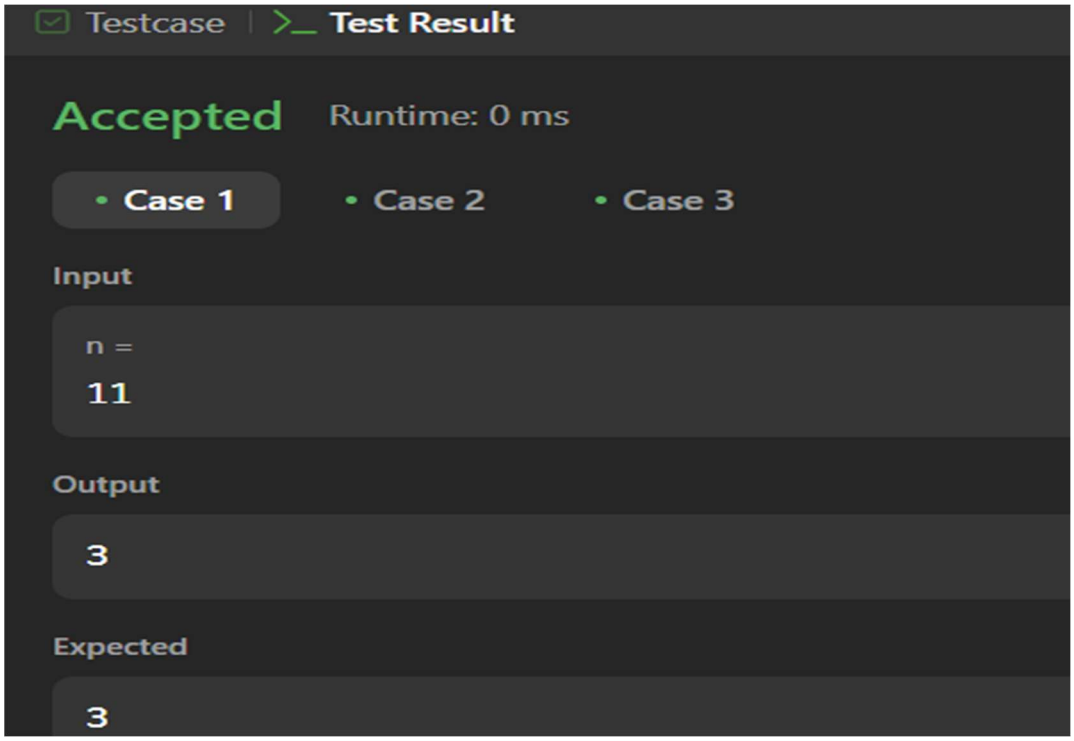
```
class Solution {
public:
    TreeNode* sortedArrayToBST(vector<int>& nums) {
        return helper(nums, 0, nums.size() - 1);
    }

    TreeNode* helper(vector<int>& nums, int left, int right) {
        if (left > right) return nullptr;
        int mid = left + (right - left) / 2;
        TreeNode* root = new TreeNode(nums[mid]);
        root->left = helper(nums, left, mid - 1);
        root->right = helper(nums, mid + 1, right);
        return root;
    }
};
```



Q2. Number of 1 Bits (LeetCode 191)

```
class Solution {
public:
    int hammingWeight(uint32_t n) {
        int count = 0;
        while (n) {
            count += n & 1;
            n >>= 1;
        }
        return count;
    }
};
```



Q3. Sort an Array (LeetCode 912)

```
class Solution {
public:
    vector<int> sortArray(vector<int>& nums) {
        sort(nums.begin(), nums.end());
        return nums;
    }
};
```

Q4. Maximum Subarray (LeetCode 53)

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = nums[0], currSum = 0;
        for (int num : nums) {
            currSum = max(num, currSum + num);
            maxSum = max(maxSum, currSum);
        }
```

```
        return maxSum;
    }
};
```

Q5. Beautiful Array (LeetCode 932)

```
class Solution {
public:
    vector<int> beautifulArray(int n) {
        vector<int> res = {1};
        while (res.size() < n) {
            vector<int> temp;
            for (int num : res) if (num * 2 - 1 <= n) temp.push_back(num * 2 - 1);
            for (int num : res) if (num * 2 <= n) temp.push_back(num * 2);
            res = temp;
        }
        return res;
    }
};
```

Q6. Super Pow (LeetCode 372)

```
class Solution {
public:
    int modPow(int a, int b, int mod) {
        int res = 1;
        a %= mod;
        while (b > 0) {
            if (b % 2 == 1) res = (long long)res * a % mod;
            a = (long long)a * a % mod;
            b /= 2;
        }
        return res;
    }

    int superPow(int a, vector<int>& b) {
        const int mod = 1337;
        int res = 1;
        for (int digit : b) {
```

```
            res = modPow(res, 10, mod) * modPow(a, digit, mod) % mod;
        }
        return res;
    }
};
```

Q7.The Skyline Problem (LeetCode 218)

```
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
        vector<pair<int, int>> events;
        for (auto& b : buildings) {
            events.emplace_back(b[0], -b[2]);  // Start of a building
            events.emplace_back(b[1], b[2]);   // End of a building
        }
        sort(events.begin(), events.end());

        multiset<int> heights = {0};
        vector<vector<int>> result;
        int prevHeight = 0;

        for (auto& [x, h] : events) {
            if (h < 0) heights.insert(-h);  // Insert start height
            else heights.erase(heights.find(h));  // Remove end height

            int currHeight = *heights.rbegin();
            if (currHeight != prevHeight) {
                result.push_back({x, currHeight});
                prevHeight = currHeight;
            }
        }
        return result;
    }
};
```