Assignment 6

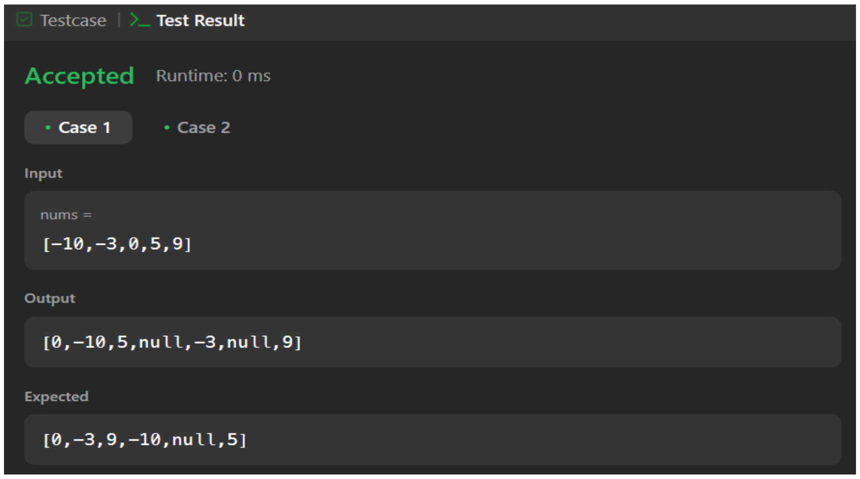
Student Name: Bhuvnesh Gautam **UID:** 22BCS14051
Branch: CSE **Section/Group:** 22BCS_IOT-611 B
Semester: 6th **Date of Performance:** 19/03/2025
Subject Name: Advanced Programming Lab - 2
Subject Code: 22CSP-351

Problem 108. Convert Sorted Array to Binary Search Tree

- **Implementation/Code:**

```
class Solution {
public:
    TreeNode* sortedArrayToBST(vector<int>& nums) {
        return buildBST(nums, 0, nums.size() - 1);
    }
private:
    TreeNode* buildBST(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 = buildBST(nums, left, mid - 1);
        root->right = buildBST(nums, mid + 1, right);
        return root;
    }
};
```

- **Output:**



Accepted Runtime: 0 ms

Input

nums =

```
[-10,-3,0,5,9]
```

Output

```
[0,-10,5,null,-3,null,9]
```

Expected

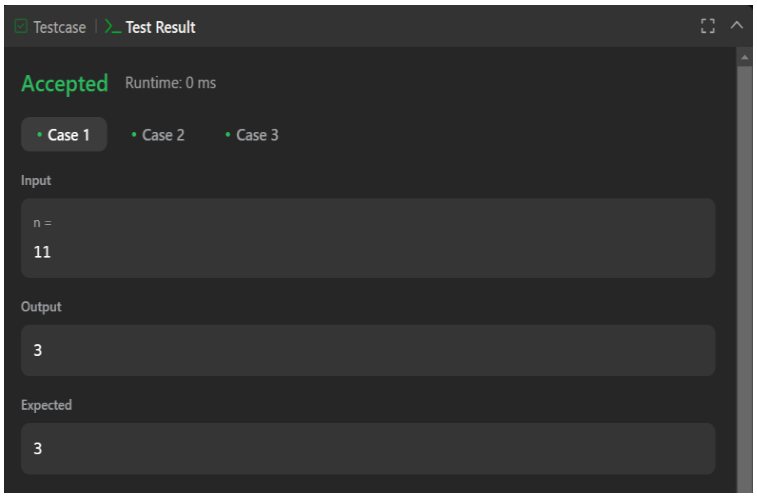
```
[0,-3,9,-10,null,5]
```

Problem 191. Number of 1 Bits

- **Implementation/Code:**

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

- **Output:**



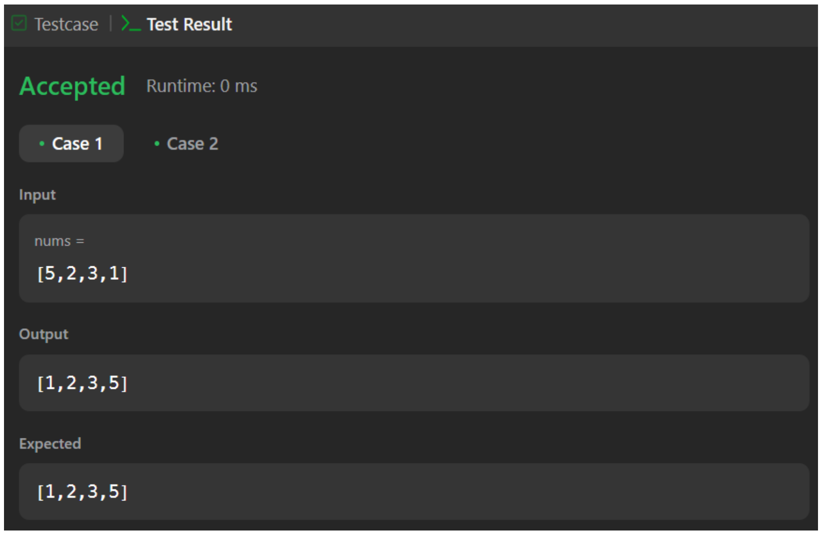
```
Testcase | Test Result

Accepted   Runtime: 0 ms

Case 1   Case 2   Case 3

Input
n =
11

Output
3

Expected
3
```

Problem 912. Sort an Array

- **Implementation/Code:**

```
class Solution {
public:
    vector<int> sortArray(vector<int>& nums) {
        mergeSort(nums, 0, nums.size() - 1);
        return nums;
    }
private:
    void mergeSort(vector<int>& nums, int left, int right) {
        if (left >= right) return;
        int mid = left + (right - left) / 2;
        mergeSort(nums, left, mid);
        mergeSort(nums, mid + 1, right);
        merge(nums, left, mid, right);
    }
    void merge(vector<int>& nums, int left, int mid, int right) {
        vector<int> temp;
        int i = left, j = mid + 1;
        while (i <= mid && j <= right) {
            if (nums[i] <= nums[j]) temp.push_back(nums[i++]);
            else temp.push_back(nums[j++]);
        }
        while (i <= mid) temp.push_back(nums[i++]);
        while (j <= right) temp.push_back(nums[j++]);
        for (int k = 0; k < temp.size(); ++k)
            nums[left + k] = temp[k];
    }
};
```

- **Output:**



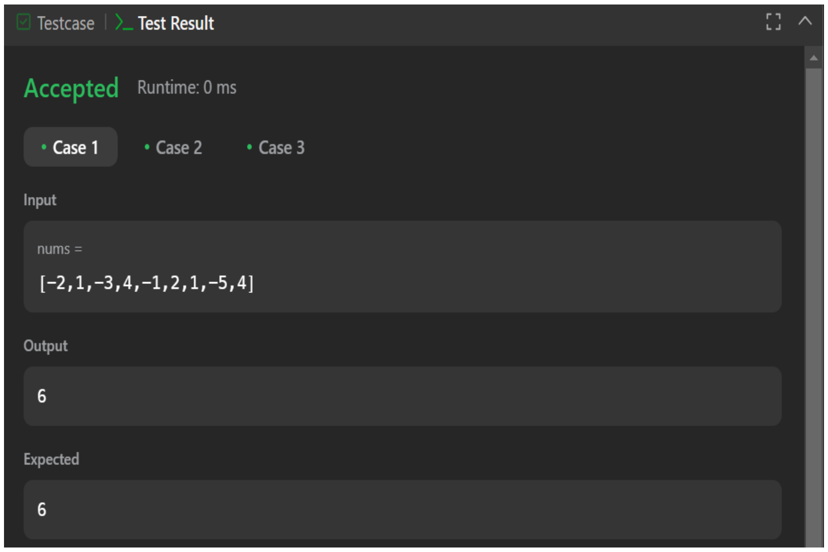
```
Accepted   Runtime: 0 ms
Input
nums =
[5,2,3,1]
Output
[1,2,3,5]
Expected
[1,2,3,5]
```

Problem 53. Maximum Subarray

- **Implementation/Code:**

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int sum = 0;
        int n = nums.size();
        int maximum = nums[0];
        for (int i = 0; i < n; i++) {
                sum += nums[i];
            maximum = max(maximum, sum);
            if (sum < 0) sum = 0;
        }
        return maximum;
    }
};
```

- **Output:**



Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums =
[-2,1,-3,4,-1,2,1,-5,4]

Output

6

Expected

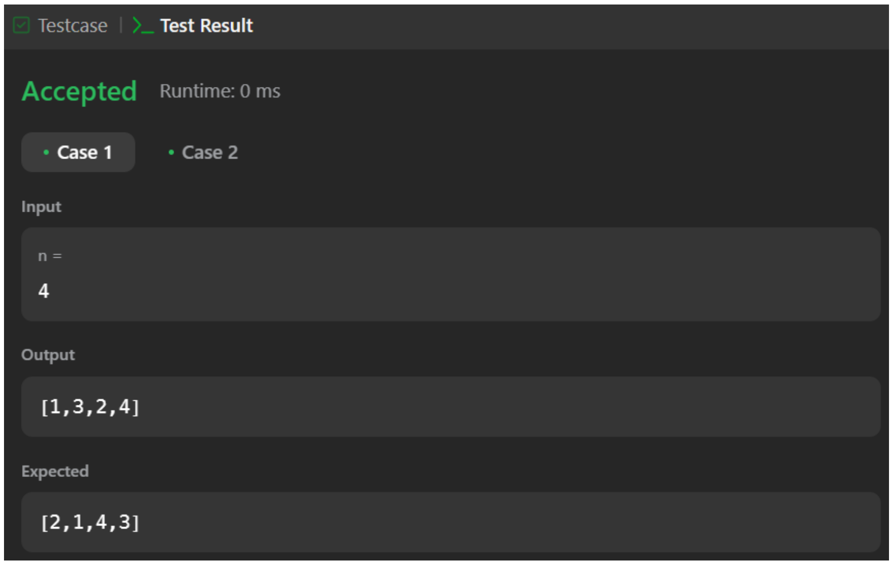
6

Problem 932. Beautiful Array

- **Implementation/Code:**

```
class Solution {
public:
    vector<int> beautifulArray(int n) {
        if (n == 1) return {1};

        vector<int> result;
        vector<int> oddPart = beautifulArray((n + 1) / 2);
        vector<int> evenPart = beautifulArray(n / 2);

        for (int num : oddPart) result.push_back(num * 2 - 1);
        for (int num : evenPart) result.push_back(num * 2);

        return result;
    }
};
```

- **Output:**



Accepted Runtime: 0 ms

Input

```
n =
4
```

Output

```
[1,3,2,4]
```

Expected

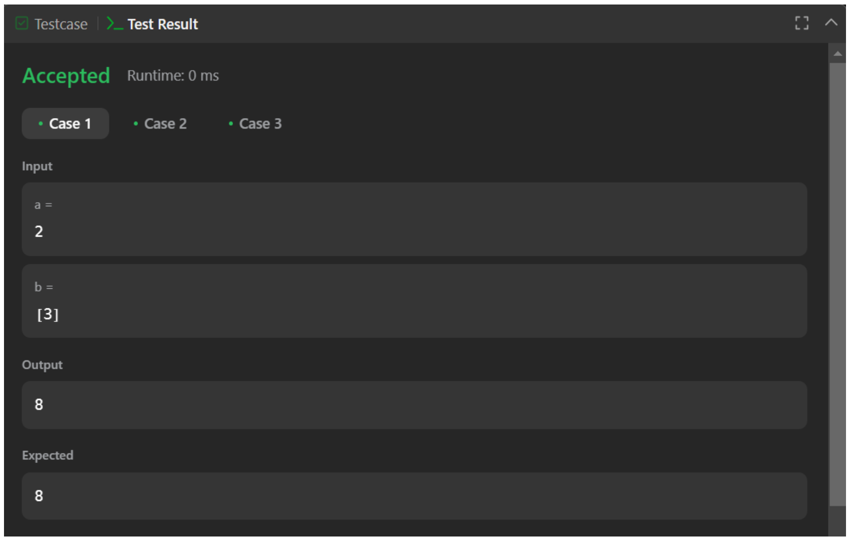
```
[2,1,4,3]
```

Problem 372. Super Pow

- **Implementation/Code:**

```
class Solution {
public:
    const int MOD = 1337;
    int powerMod(int a, int k) {
        a %= MOD;
        int res = 1;
        while (k > 0) {
            if (k % 2 == 1) {
                res = (res * a) % MOD;
            }
            a = (a * a) % MOD;
            k /= 2;
        }
        return res;
    }

    int superPow(int a, vector<int>& b) {
        int result = 1;
        for (int digit : b) {
            result = powerMod(result, 10) * powerMod(a, digit) % MOD;
        }
        return result;
    }
};
```

- **Output:**



```
Accepted   Runtime: 0 ms

Input
a =
2
b =
[3]
Output
8
Expected
8
```

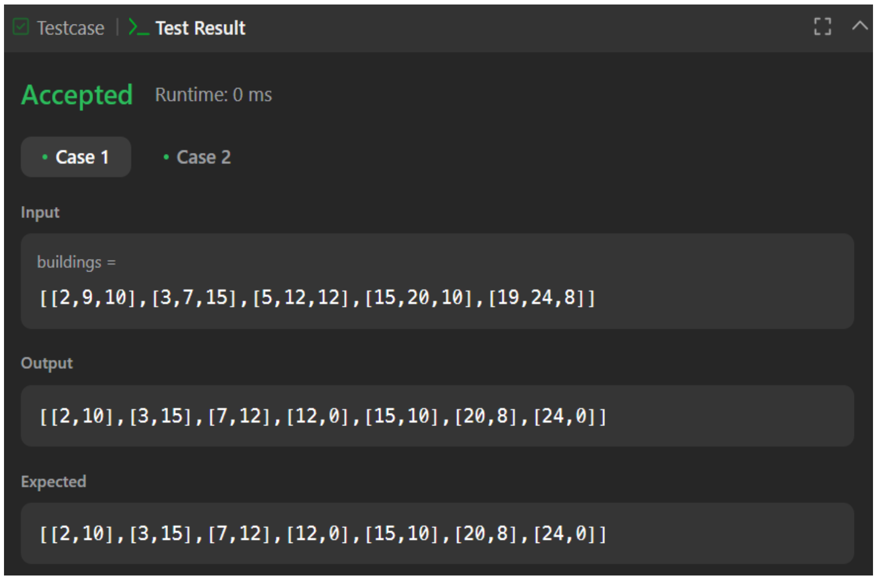
Problem 218. The Skyline Problem

- **Implementation/Code:**

```
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]);
            events.emplace_back(b[1], b[2]);
        }

        sort(events.begin(), events.end(), [](const pair<int, int>& a, const pair<int, int>& b) {
            if (a.first != b.first) return a.first < b.first;
            return a.second < b.second;
        });

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

        for (auto& [x, h] : events) {
            if (h < 0) {
                heights.insert(-h);
            } else {
                heights.erase(heights.find(h));
            }

            int curMax = *heights.rbegin();
```

```
                if (curMax != prevMax) {
                    result.push_back({x, curMax});
                    prevMax = curMax;
                }
            }
            return result;
        }
    };
```

- **Output:**



Accepted Runtime: 0 ms

Input

buildings =

```
[[2,9,10],[3,7,15],[5,12,12],[15,20,10],[19,24,8]]
```

Output

```
[[2,10],[3,15],[7,12],[12,0],[15,10],[20,8],[24,0]]
```

Expected

```
[[2,10],[3,15],[7,12],[12,0],[15,10],[20,8],[24,0]]
```