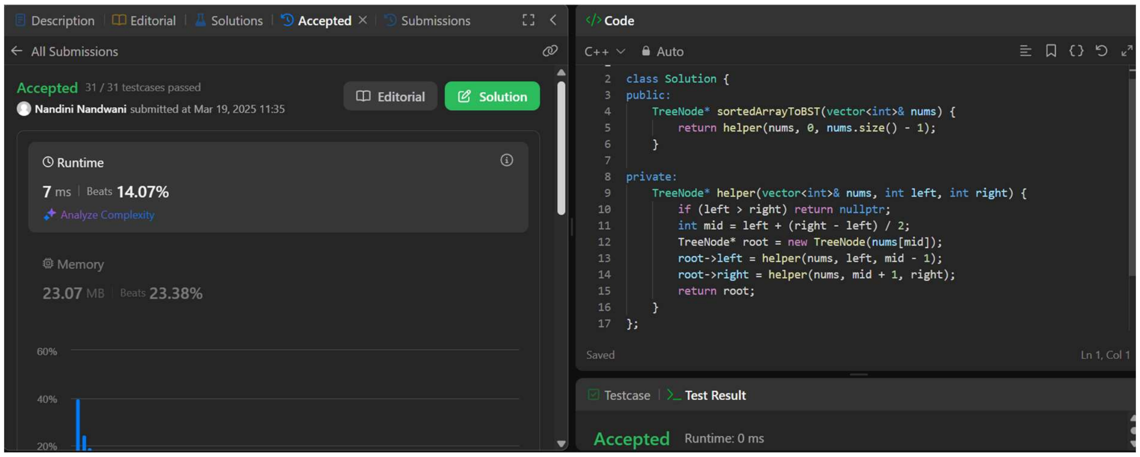
Nandini Nandwani

22BCS11637

108. Convert Sorted Array to Binary Search Tree



Accepted 31 / 31 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:35

Runtime
7 ms | Beats 14.07%

Memory
23.07 MB | Beats 23.38%

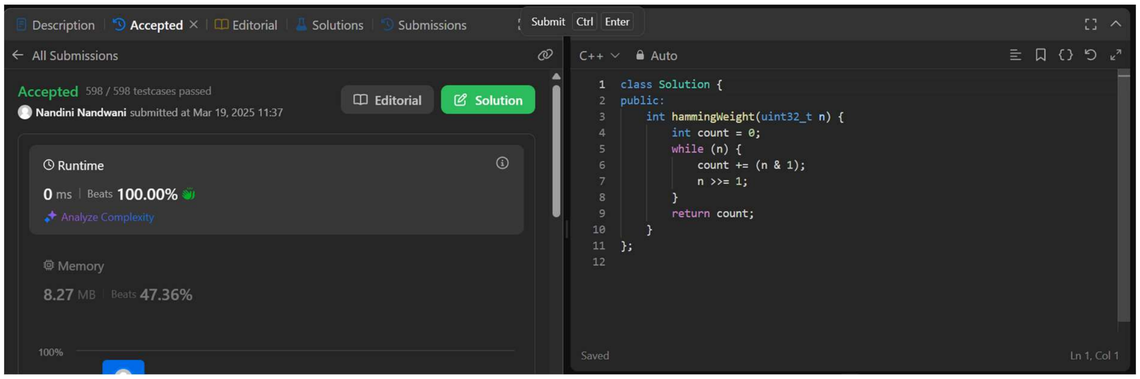
```cpp
class Solution {
public:
    TreeNode* sortedArrayToBST(vector<int>& nums) {
        return helper(nums, 0, nums.size() - 1);
    }

private:
    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;
    }
};
```

Saved Ln 1, Col 1

Accepted Runtime: 0 ms

191. Number of 1 Bits



Accepted 598 / 598 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:37

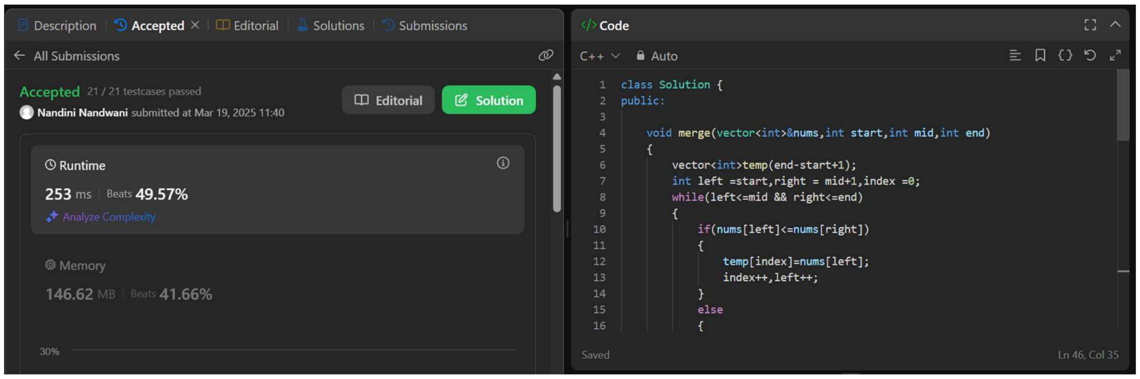
Runtime
0 ms | Beats 100.00%

Memory
8.27 MB | Beats 47.36%

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

Saved Ln 1, Col 1

912. Sort an Array



Accepted 21 / 21 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:40

Runtime
253 ms | Beats 49.57%

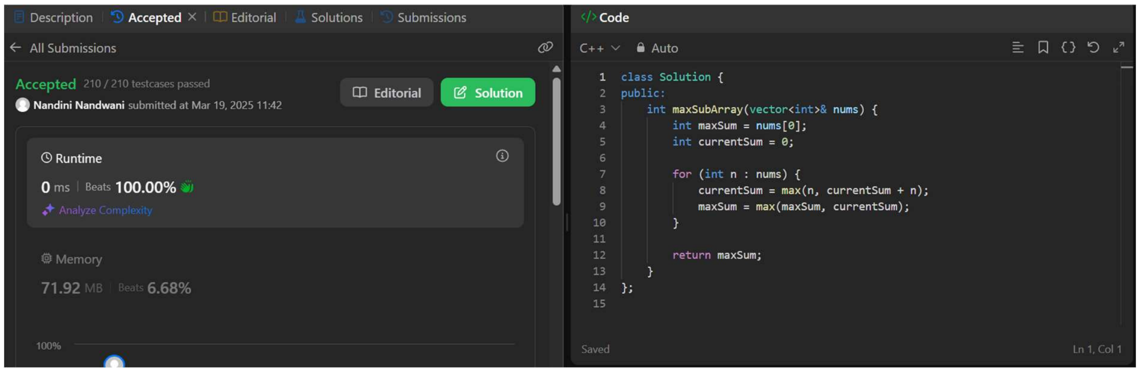
Memory
146.62 MB | Beats 41.66%

```cpp
class Solution {
public:

    void merge(vector<int>&nums,int start,int mid,int end)
    {
        vector<int>temp(end-start+1);
        int left =start,right = mid+1,index =0;
        while(left<=mid && right<=end)
        {
            if(nums[left]<=nums[right])
            {
                temp[index]=nums[left];
                index++,left++;
            }
            else
            {
```

Saved Ln 46, Col 35

53.Maximum Subarray


Accepted 210 / 210 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:42

Runtime
0 ms | Beats 100.00%

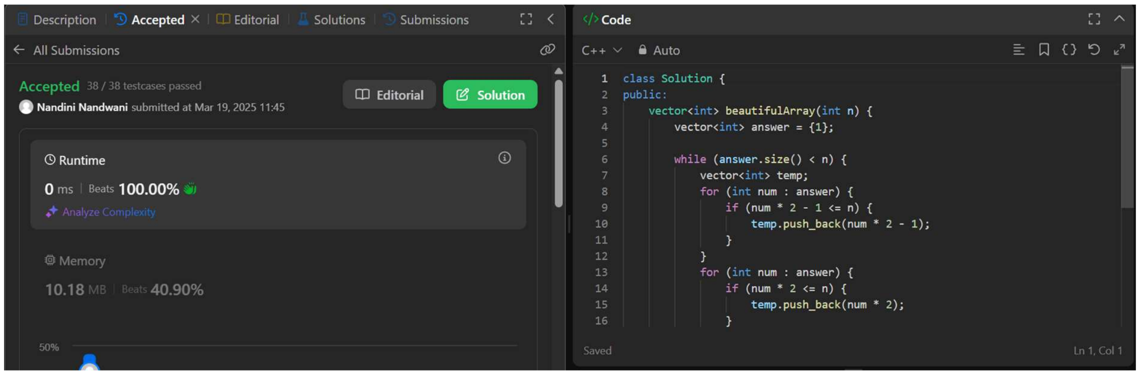
Memory
71.92 MB | Beats 6.68%

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = nums[0];
        int currentSum = 0;

        for (int n : nums) {
            currentSum = max(n, currentSum + n);
            maxSum = max(maxSum, currentSum);
        }

        return maxSum;
    }
};
```

932.Beautiful Array


Accepted 38 / 38 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:45

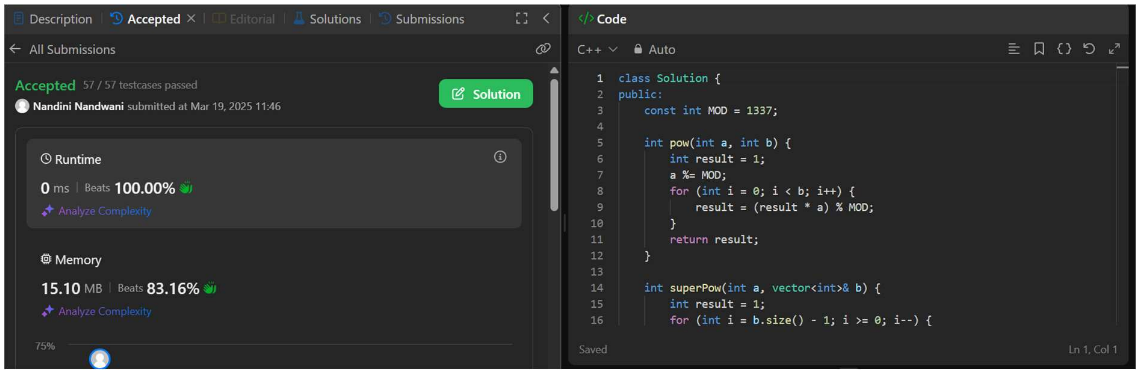
Runtime
0 ms | Beats 100.00%

Memory
10.18 MB | Beats 40.90%

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

        while (answer.size() < n) {
            vector<int> temp;
            for (int num : answer) {
                if (num * 2 - 1 <= n) {
                    temp.push_back(num * 2 - 1);
                }
            }
            for (int num : answer) {
                if (num * 2 <= n) {
                    temp.push_back(num * 2);
                }
```

372.Super Pow


Accepted 57 / 57 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:46

Runtime
0 ms | Beats 100.00%

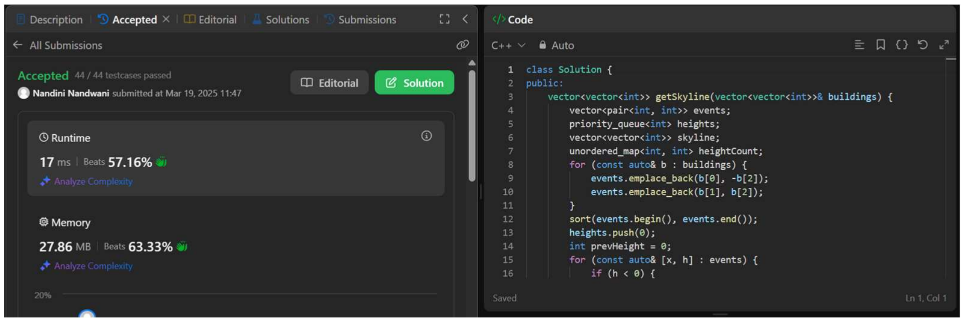
Memory
15.10 MB | Beats 83.16%

```
class Solution {
public:
    const int MOD = 1337;

    int pow(int a, int b) {
        int result = 1;
        a %= MOD;
        for (int i = 0; i < b; i++) {
            result = (result * a) % MOD;
        }
        return result;
    }

    int superPow(int a, vector<int>& b) {
        int result = 1;
        for (int i = b.size() - 1; i >= 0; i--) {
```

218.The Skyline Problem


Accepted 44 / 44 testcases passed
Nandini Nandwani submitted at Mar 19, 2025 11:47

Runtime
17 ms | Beats 57.16%

Memory
27.86 MB | Beats 63.33%

```
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
        vector<pair<int, int>> events;
        priority_queue<int> heights;
        vector<vector<int>> skyline;
        unordered_map<int, int> heightCount;
        for (const auto& b : buildings) {
            events.emplace_back(b[0], -b[2]);
            events.emplace_back(b[1], b[2]);
        }
        sort(events.begin(), events.end());
        heights.push(0);
        int prevHeight = 0;
        for (const auto& [x, h] : events) {
            if (h < 0) {
```