

## Assignment 6

### 108.Convert Sorted Array to Binary Search Tree

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

    TreeNode* buildBST(vector<int>& nums, int start, int end) {
        if (start > end) return NULL;

        int mid = start + (end - start) / 2;
        TreeNode* root = new TreeNode(nums[mid]);

        root->left = buildBST(nums, start, mid - 1);
        root->right = buildBST(nums, mid + 1, end);

        return root;
    }
};
```

The screenshot displays a coding platform interface for the problem '108.Convert Sorted Array to Binary Search Tree'. The left sidebar shows the problem status as 'Accepted' with 31/31 testcases passed, submitted by 'Ashish Kum...' on Mar 19, 2025 at 11:52. The runtime is 3 ms, beating 60.10% of other solutions, and memory usage is 23.04 MB, beating 23.38%. A bar chart shows the distribution of runtime results. The main area displays the C++ code for the solution, which is a recursive function to build a BST from a sorted array. The right sidebar shows the test result as 'Accepted' with a runtime of 0 ms.

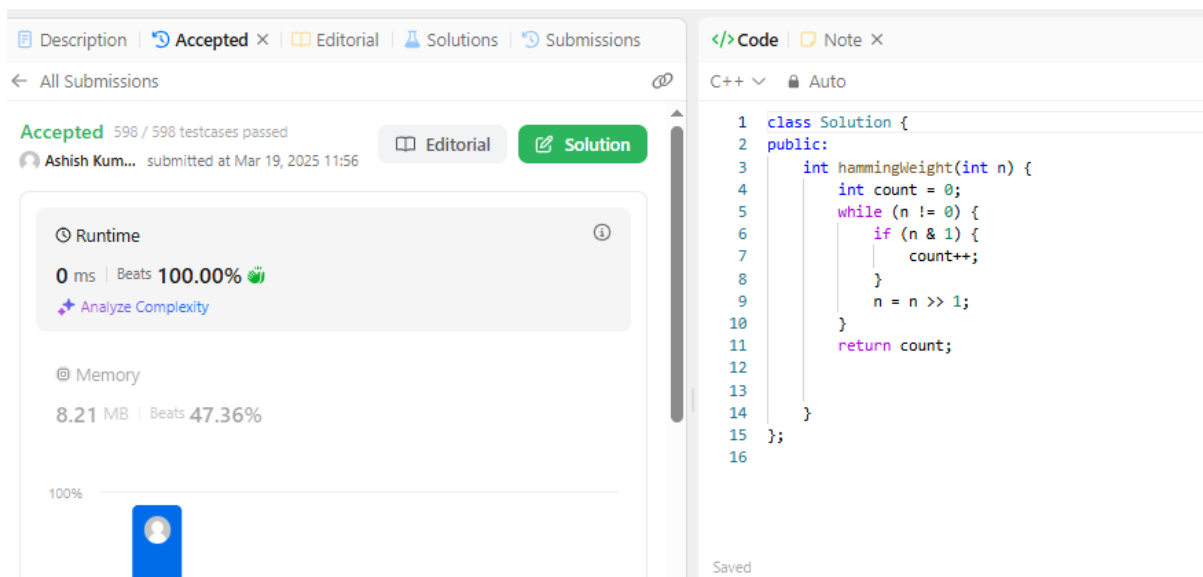
### 191.Number of 1 Bits

```
class Solution {
public:
```

```

int hammingWeight(int n) {
    int count = 0;
    while (n != 0) {
        if (n & 1) {
            count++;
        }
        n = n >> 1;
    }
    return count;
}
};

```



Accepted 596 / 598 testcases passed

Ashish Kum... submitted at Mar 19, 2025 11:56

Runtime: 0 ms | Beats 100.00%

Memory: 8.21 MB | Beats 47.36%

```

1 class Solution {
2 public:
3     int hammingWeight(int n) {
4         int count = 0;
5         while (n != 0) {
6             if (n & 1) {
7                 count++;
8             }
9             n = n >> 1;
10        }
11        return count;
12    }
13 }
14 };
15
16

```

## 912. Sort an Array

```

#include <vector>

using namespace std;

class Solution {
public:
    void merge(vector<int>& nums, int left, int mid, int right) {
        int n1 = mid - left + 1, n2 = right - mid;
        vector<int> L(n1), R(n2);
        for (int i = 0; i < n1; i++) L[i] = nums[left + i];
        for (int i = 0; i < n2; i++) R[i] = nums[mid + 1 + i];
    }
};

```

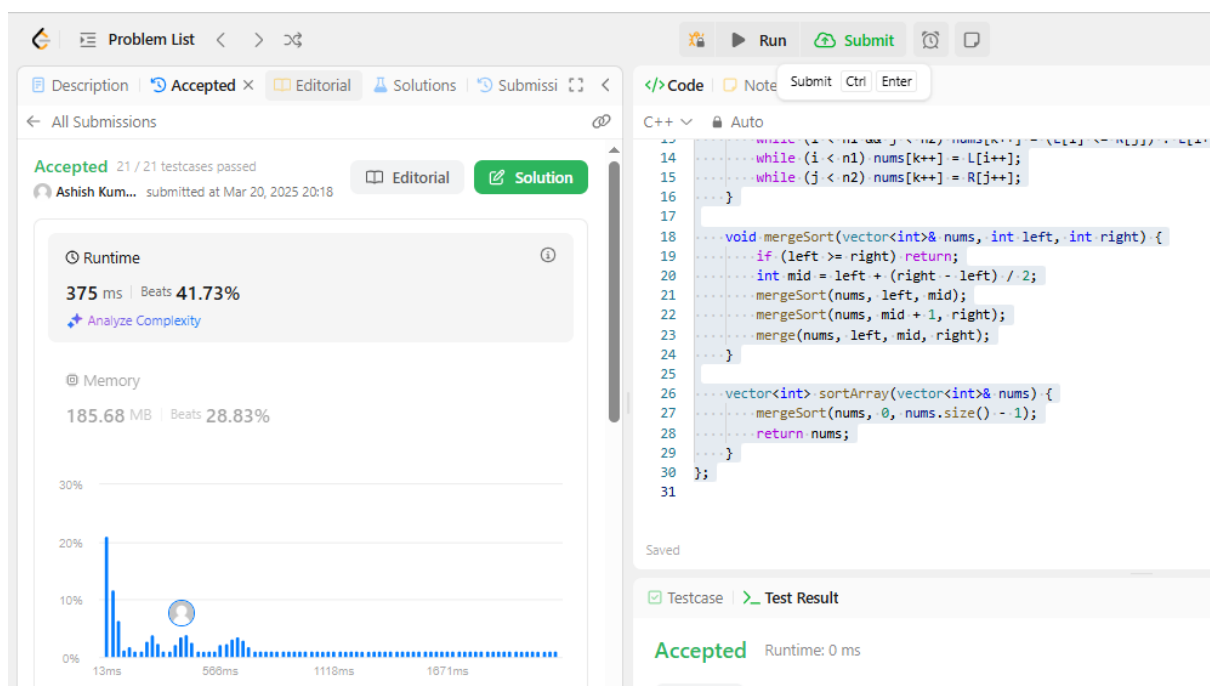
```

    int i = 0, j = 0, k = left;
    while (i < n1 && j < n2) nums[k++] = (L[i] <= R[j]) ? L[i++] : R[j++];
    while (i < n1) nums[k++] = L[i++];
    while (j < n2) nums[k++] = R[j++];
}

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);
}

vector<int> sortArray(vector<int>& nums) {
    mergeSort(nums, 0, nums.size() - 1);
    return nums;
}
};

```



## 53. Maximum Subarray

```

class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = INT_MIN, currSum = 0;

        for (int num : nums) {
            currSum = max(num, currSum + num);

```

```

        maxSum = max(maxSum, currSum);
    }

    return maxSum;
}
};

```

The screenshot displays a LeetCode submission interface. On the left, the 'Accepted' status is confirmed with 210/210 testcases passed. Performance metrics show a runtime of 0 ms, beating 100.00% of submissions, and a memory usage of 71.71 MB, beating 53.31%. A bar chart visualizes the user's performance against other participants. The code editor on the right contains the following C++ code:

```

1 class Solution {
2 public:
3     int maxSubArray(vector<int>& nums) {
4         int maxSum = INT_MIN, currSum = 0;
5
6         for (int num : nums) {
7             currSum = max(num, currSum + num);
8             maxSum = max(maxSum, currSum);
9         }
10
11        return maxSum;
12    }
13 };
14
15

```

## 932. Beautiful Array

```
#include <vector>
```

```
using namespace std;
```

```
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;

}

};

```

Accepted 38 / 38 testcases passed  
Ashish Kum... submitted at Mar 20, 2025 20:20

Runtime: 3 ms | Beats 38.58%  
Memory: 10.18 MB | Beats 39.92%

```

1 #include <vector>
2
3 using namespace std;
4
5 class Solution {
6 public:
7     vector<int> beautifulArray(int n) {
8         vector<int> res = {1};
9         while (res.size() < n) {
10             vector<int> temp;
11             for (int num : res) if (num * 2 - 1 <= n) temp.push_back(num * 2 - 1);
12             for (int num : res) if (num * 2 <= n) temp.push_back(num * 2);
13             res = temp;
14         }
15         return res;
16     }
17 };
18

```

## 372. Super Pow

```

class Solution {
public:
    const int MOD = 1337;

    int modPower(int a, int k) {
        int result = 1;
        a %= MOD;
        while (k > 0) {
            if (k % 2 == 1) result = (result * a) % MOD;
            a = (a * a) % MOD;
            k /= 2;
        }
        return result;
    }

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

```

**Accepted** 57 / 57 testcases passed  
 Ashish Kumar Saurav submitted at Mar 20, 2025 20:22

**Runtime**  
 0 ms | Beats 100.00%  
 Analyze Complexity

**Memory**  
 15.20 MB | Beats 50.99%

**Code**

```

1 class Solution {
2 public:
3     const int MOD = 1337;
4
5     int modPower(int a, int k) {
6         int result = 1;
7         a %= MOD;
8         while (k > 0) {
9             if (k % 2 == 1) result = (result * a) % MOD;
10            a = (a * a) % MOD;
11            k /= 2;
12        }
13        return result;
14    }
15
16    int superPow(int a, vector<int>& b) {
17        int result = 1;
18        for (int digit : b) {
19            result = modPower(result, 10) * modPower(a, digit) % MOD;
20        }
21    }
22 }
    
```

Testcase Test Result

Case 1 Case 2 Case 3 +

## 218. The Skyline Problem

```
#include <vector>
```

```
#include <set>
```

```
#include <map>
```

```
using namespace std;
```

```
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());
```

```
    multiset<int> heights = {0};
```

```
    vector<vector<int>> result;
```

```

int prevHeight = 0;

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

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

return result;
}
};

```

**Problem List**

[Description](#)
[Accepted](#)
[Editorial](#)
[Solutions](#)
[Submissions](#)

All Submissions

Accepted 44 / 44 testcases passed  
Ashish Kum... submitted at Mar 20, 2025 20:24

Runtime  
15 ms | Beats 65.81%  
[Analyze Complexity](#)

Memory  
27.65 MB | Beats 75.69%

Code
Note

```

17 //vector<vector<int>> heights;
18 vector<vector<int>> result;
19 int prevHeight = 0;
20
21 for (auto& [x, h] : events) {
22     if (h < 0) heights.insert(-h);
23     else heights.erase(heights.find(h));
24
25     int currHeight = *heights.rbegin();
26     if (currHeight != prevHeight) {
27         result.push_back({x, currHeight});
28         prevHeight = currHeight;
29     }
30 }
31
32 return result;
33 }
34 };
35

```

Testcase
Test Result

Accepted Runtime: 2 ms  
Case 1 Case 2