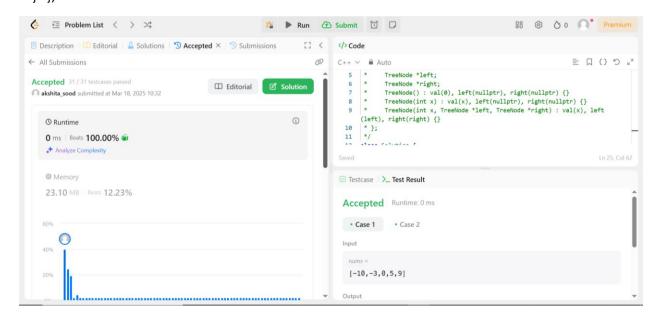
Name: Akshita Sood Uid: 22BCS14993 Section: FL_lot 601 'A'

1.Convert Sorted Array to Binary Search Tree

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



2. Number of 1 Bits class Solution {

```
public:
                    int hammingWeight(int n) {
                                    int count = 0;
                                  for(int i = 31; i >= 0; i--){
                                                  if(((n >> i) \& 1) == 1)
                                                                 count++;
                                    return count;
                                                                                                                                                                                                                                                                                                                                                                                                                                                BB S O Premium
♦ Problem List 〈 > >औ
                                                                                                                                                                                                                 🔏 🕨 Run 🚹 Submit 🔯 🗔
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     [] ^
                                                                                                                                                                                                                                   [] < </>Code

□ Description □ Editorial □ Accepted □ Submissions
□ Description □ Editorial □ Editorial □ □ Solutions □ □ Accepted □ □ Submissions
□ Description □ □ Editorial □ Editori
← All Submissions
                                                                                                                                                                                                                                                                                 ™ C {} □ ≡
                                                                                                                                                                                                                                                                                          1 class Solution {
  Accepted 598 / 598 testcases passed
                                                                                                                                                                                                                                                                                                                 public:
  akshita_sood submitted at Mar 18, 2025 10:43
                                                                                                                                                                                                                                                                                                                               int hammingWeight(int n) {
                                                                                                                                                                                                                                                                                                                                           int count = 0;
for(int i = 31; i >= 0; i--){
   if(((n >> i) & 1) == 1)
            O Runtime
              0 ms | Beats 100.00% 🦥
             Analyze Complexity
                                                                                                                                                                                                                                                                                   ☑ Testcase | >_ Test Result
             8.16 MB | Beats 80.26% 🞳
                                                                                                                                                                                                                                                                                      Accepted Runtime: 0 ms
                                                                                                                                                                                                                                                                                       • Case 1 • Case 2 • Case 3
                                                                                                                                                                                                                                                                                          11
```

3. Sort an Array

```
class Solution {
  public:
  vector<int> sortArray(vector<int>& nums) {
    mergeSort(nums, 0, nums.size() - 1);
    return nums;
  }
  private:
  void mergeSort(vector<int>& nums, int I, int r) {
    if (I >= r)
      return;
    const int m = (I + r) / 2;
    mergeSort(nums, I, m);
  }
}
```

```
mergeSort(nums, m + 1, r);
 merge(nums, I, m, r);
void merge(vector<int>& nums, int I, int m, int r) {
 vector<int> sorted(r - l + 1);
 int k = 0;
 int i = I;
 int j = m + 1;
 while (i <= m \&\& j <= r)
  if (nums[i] < nums[j])</pre>
    sorted[k++] = nums[i++];
   else
    sorted[k++] = nums[j++];
 while (i \le m)
   sorted[k++] = nums[i++];
 while (j \le r)
   sorted[k++] = nums[j++];
 copy(sorted.begin(), sorted.end(), nums.begin() + I);
                                             🌋 🕨 Run 🚹 Submit 🔯 🔘
♦ E Problem List < > >
🖪 Description | 🕮 Editorial | 🚨 Solutions | 🤊 Accepted × | 🧐 Submissions
                                                                                                        『』 で ( ) 口 重
← All Submissions
                                                           C++ ∨ Auto
                                                             1 class Solution {
Accepted 21 / 21 testcases passed
                                                                  public:
akshita_sood submitted at Mar 18, 2025 10:51
                                                                    vector<int> sortArray(vector<int>& nums) {
                                                                       mergeSort(nums, θ, nums.size() - 1);
return nums;
                                                  (i)
   O Runtime
   248 ms | Beats 50.29% 🞳
                                                                  private:
   ♣ Analyze Complexity
                                                           Saved & Upgrade to Cloud Saving
   @ Memory
                                                            146.69 MB | Beats 41.83%
                                                            Accepted Runtime: 0 ms
                                                            • Case 1 • Case 2
                                                            Input
                                                             [5,2,3,1]
          0
```

4. Maximum Subarray

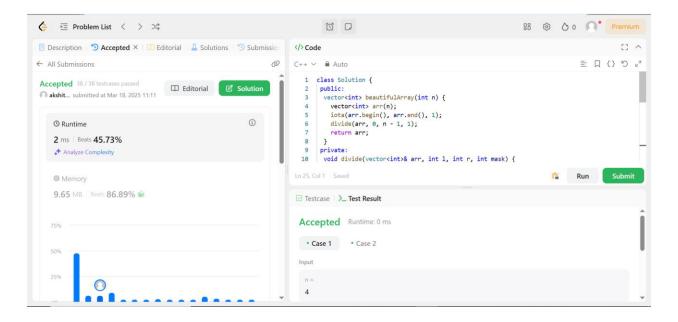
```
class Solution {
  public:
     int maxSubArray(vector<int>& nums) {
          vector<int> dp(nums.size());
           dp[0] = nums[0];
           for (int i = 1; i < nums.size(); ++i)
                 dp[i] = max(nums[i], dp[i - 1] + nums[i]);
           return ranges::max(dp);
     }
};
                                                                                                                                                                                                                                                                                                                                                                     ₽8 ⊗ O O Premium
       ♦ Problem List < > >
                                                                                                                                                                                                                       0 0

■ Description  
■ Accepted ×  
■ Editorial  
■ Solutions  
■ S  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ C  
■ 
                                                                                                                                                                                                </>Code
                                                                                                                                                                                                                                                                                                                                                                                                                                           [] ^
                                                                                                                                                                                                                                                                                                                                                                                                              ™ C () □ ≡
                                                                                                                                                                                                C++ ∨ 🗎 Auto
                                                                                                                                                                                                       4 #include <limits.h> // For INT_MIN
         Accepted 210 / 210 testcases passed
                                                                                                                                                                                                       5 #include <vector>
          akshit... submitted at Mar 18, 2025 11:08
                                                                                                                                                                                                               class Solution {
                                                                                                                                                                                                                         int maxSubArray(std::vector<int>& nums) {
  int maxSum = INT_MIN, currSum = 0;
                                                                                                                                                               (i)
                  O Runtime
                                                                                                                                                                                                                                    for (int num : nums) {
   currSum = std::max(num, currSum + num);
                  0 ms | Beats 100.00% 🞳
                  Analyze Complexity
                                                                                                                                                                                                                                              maxSum = std::max(maxSum, currSum);
                  71.63 MB | Beats 81.06% 🞳
                                                                                                                                                                                                 Accepted Runtime: 0 ms
                                        0
                                                                                                                                                                                                    • Case 1 • Case 2 • Case 3
                                                                                                                                                                                                       [-2,1,-3,4,-1,2,1,-5,4]
```

5. Beautiful Array

```
class Solution {
  public:
  vector<int> beautifulArray(int n) {
    vector<int> arr(n);
    iota(arr.begin(), arr.end(), 1);
    divide(arr, 0, n - 1, 1);
    return arr;
  }
  private:
  void divide(vector<int>& arr, int I, int r, int mask) {
    if (I >= r)
      return;
  }
}
```

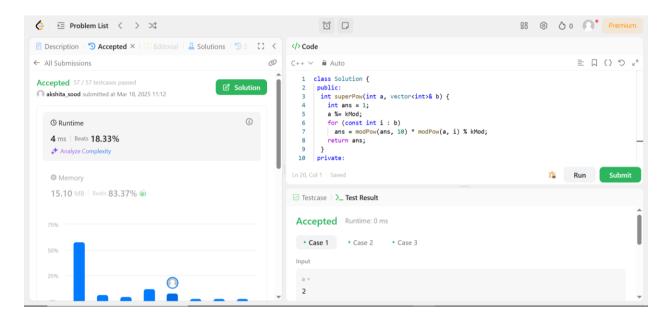
```
const int m = partition(arr, l, r, mask);
  divide(arr, l, m, mask << 1);
  divide(arr, m + 1, r, mask << 1);
}
int partition(vector<int>& arr, int l, int r, int mask) {
  int nextSwapped = l;
  for (int i = l; i <= r; ++i)
    if (arr[i] & mask)
      swap(arr[i], arr[nextSwapped++]);
  return nextSwapped - 1;
}
</pre>
```



6. Super Pow

```
class Solution {
  public:
  int superPow(int a, vector<int>& b) {
    int ans = 1;
    a %= kMod;
    for (const int i : b)
      ans = modPow(ans, 10) * modPow(a, i) % kMod;
    return ans;
  }
  private:
```

```
static constexpr int kMod = 1337;
long modPow(long x, long n) {
  if (n == 0)
    return 1;
  if (n % 2 == 1)
    return x * modPow(x % kMod, (n - 1)) % kMod;
  return modPow(x * x % kMod, (n / 2)) % kMod;
}
```



7. The Skyline Problem

```
class Solution {
  public:
  vector<vector<int>>> getSkyline(const vector<vector<int>>& buildings) {
    const int n = buildings.size();
    if (n == 0)
      return {};
    if (n == 1) {
      const int left = buildings[0][0];
      const int right = buildings[0][1];
      const int height = buildings[0][2];
      return {{left, height}, {right, 0}};
    }
```

```
const vector<vector<int>> left =
    getSkyline({buildings.begin(), buildings.begin() + n / 2});
 const vector<vector<int>> right =
    getSkyline({buildings.begin() + n / 2, buildings.end()});
 return merge(left, right);
}
private:
vector<vector<int>> merge(const vector<vector<int>>& left,
                const vector<vector<int>>& right) {
 vector<vector<int>> ans;
 int i = 0;
 int i = 0;
 int leftY = 0;
 int rightY = 0;
 while (i < left.size() && j < right.size())
  if (left[i][0] < right[j][0]) {
    leftY = left[i][1];
    addPoint(ans, left[i][0], max(left[i++][1], rightY));
  } else {
    rightY = right[j][1];
    addPoint(ans, right[j][0], max(right[j++][1], leftY));
  }
 while (i < left.size())
  addPoint(ans, left[i][0], left[i++][1]);
 while (j < right.size())
  addPoint(ans, right[j][0], right[j++][1]);
 return ans;
void addPoint(vector<vector<int>>& ans, int x, int y) {
 if (!ans.empty() \&\& ans.back()[0] == x) {
  ans.back()[1] = y;
```

```
return;
}
if (!ans.empty() && ans.back()[1] == y)
  return;
  ans.push_back({x, y});
}
```

