# Assignment 6

## Student Name: Diksha Pandey UID:22BCS10359

## Branch: BE-CSE (General) Section/Group: FL\_IOT-602-A

## Semester:6th Date of Performance: 18-03-25

## Subject Name: Advanced Programming Lab-2 Subject Code: 22CSP-351

# **Aim:** [108. Convert Sorted Array to Binary Search Tree](https://leetcode.com/problems/convert-sorted-array-to-binary-search-tree/)

# Implementation/ Code:

# #include <vector>

# using namespace std;

# 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;

# }

# };

# Output:Screenshot 2025-03-19 at 2.10.39 PM.png

# **Aim:** [191. Number of 1 Bits](https://leetcode.com/problems/number-of-1-bits/)

# Implementation/ Code:

# class Solution {

# public:

# int hammingWeight(uint32\_t n) {

# int res = 0;

# for (int i = 0; i < 32; i++) {

# if ((n >> i) & 1) {

# res += 1;

# }

# }

# return res;

# }

# };

# Output:Screenshot 2025-03-19 at 2.12.14 PM.png

# **Aim:** [912. Sort an Array](https://leetcode.com/problems/sort-an-array/)

# Implementation/ Code:

# class Solution {

# public:

# void merge(vector<int>& nums,int s, int e){

# int m=(s+e)/2;

# vector<int>first(m-s+1),second(e-m);

# for (int i=0;i<first.size();i++){first[i]=nums[s+i];}

# for (int i=0;i<second.size();i++){second[i]=nums[m+1+i];}

# int i1=0,i2=0,maindex=s;

# while (i1<first.size() && i2<second.size()){

# if (first[i1]<second[i2]){nums[maindex++]=first[i1++];}

# else {nums[maindex++]=second[i2++];}

# }

# while (i1<first.size()){nums[maindex++]=first[i1++];}

# while (i2<second.size()){nums[maindex++]=second[i2++];}

# }

# void mergesort(vector<int>&nums, int s, int e){

# if (s>=e){return ;}

# int m=(s+e)/2;

# mergesort(nums,s,m);

# mergesort(nums,m+1,e);

# merge(nums,s,e);

# }

# vector<int> sortArray(vector<int>& nums) {

# mergesort(nums,0,nums.size()-1);

# return nums;

# }

# };

# Output:Screenshot 2025-03-19 at 2.14.14 PM.png

# **Aim:** [53. Maximum Subarray](https://leetcode.com/problems/maximum-subarray/)

# Implementation/ Code:

# class Solution {

# public:

# int maxSubArray(vector<int>& nums) {

# int res = nums[0];

# int total = 0;

# for (int n : nums) {

# if (total < 0) {

# total = 0;

# }

# total += n;

# res = max(res, total);

# }

# return res;

# }

# };

# Output:Screenshot 2025-03-19 at 2.15.20 PM.png

# **Aim:**[932. Beautiful Array](https://leetcode.com/problems/beautiful-array/)

# Implementation/ Code:

# class Solution {

# public:

# static bool comp(const int &a, const int &b){

# int mask = 1;

# while(true)

# if((a&mask) == (b&mask)) mask = mask<<1;

# else return (a&mask) > (b&mask);

# }

# 

# vector<int> beautifulArray(int n) {

# vector<int> answer;

# while(n) answer.push\_back(n--);

# 

# sort(answer.begin(), answer.end(), comp);

# 

# return answer;

# }

# };

# Output:Screenshot 2025-03-19 at 2.16.33 PM.png

# **Aim:** [102. Binary Tree Level Order Traversal](https://leetcode.com/problems/binary-tree-level-order-traversal/)

# Implementation/ Code:

# class Solution {

# public:

# vector<vector<int>> levelOrder(TreeNode\* root) {

# if (root == nullptr)

# return {};

# vector<vector<int>> ans;

# queue<TreeNode\*> q{{root}};

# while (!q.empty()) {

# vector<int> currLevel;

# for (int sz = q.size(); sz > 0; --sz) {

# TreeNode\* node = q.front();

# q.pop();

# currLevel.push\_back(node->val);

# if (node->left)

# q.push(node->left);

# if (node->right)

# q.push(node->right);

# }

# ans.push\_back(currLevel);

# }

# return ans;

# }

# };

# Output:Screenshot 2025-02-14 at 2.15.42 PM.png

# **Aim:** [372. Super Pow](https://leetcode.com/problems/super-pow/)

# Implementation/ Code:

# class Solution {

# private:

# int solve(int base, int power, int mod) {

# int ans = 1;

# while (power > 0) {

# if (power & 1) {

# ans = (ans \* base) % mod;

# }

# base = (base \* base) % mod;

# power >>= 1;

# }

# return ans;

# }

# public:

# int superPow(int a, vector<int>& b) {

# a%=1337;

# int n = b.size();

# int m = 1140;

# int expi = 0;

# for(int i : b){

# expi = (expi\*10+i)%m;

# }

# if (expi == 0) {

# expi = m;

# }

# return solve(a,expi,1337);

# }

# };

# Output:Screenshot 2025-03-19 at 2.18.24 PM.png

# **Aim:** [218. The Skyline Problem](https://leetcode.com/problems/the-skyline-problem/)

# Implementation/ Code:

# class Solution {

# public:

# vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {

# vector<pair<int, int>> h;

# 

# for (auto b : buildings) {

# h.push\_back({b[0], -b[2]});

# h.push\_back({b[1], b[2]});

# }

# 

# 

# sort(h.begin(), h.end());

# int prev = 0, cur = 0;

# 

# multiset<int> m;

# vector<vector<int>> res;

# 

# m.insert(0);

# for (auto i:h) {

# 

# 

# if (i.second < 0) {

# m.insert(-i.second);

# } else {

# m.erase(m.find(i.second));

# }

# 

# cur = \*m.rbegin();

# 

# if (cur != prev) {

# res.push\_back({i.first, cur});

# prev = cur;

# }

# }

# return res;

# }

# };

# Output:Screenshot 2025-03-19 at 2.24.56 PM.png