Assignment -06

Student Name: Nikhil Kumar Tiwari **UID:** 22BCS10471

Branch: BE-CSE **Section/Group:** 22BCS-IOT-FL-601 A

Semester: 6th **Subject Code:** 22CSP-351

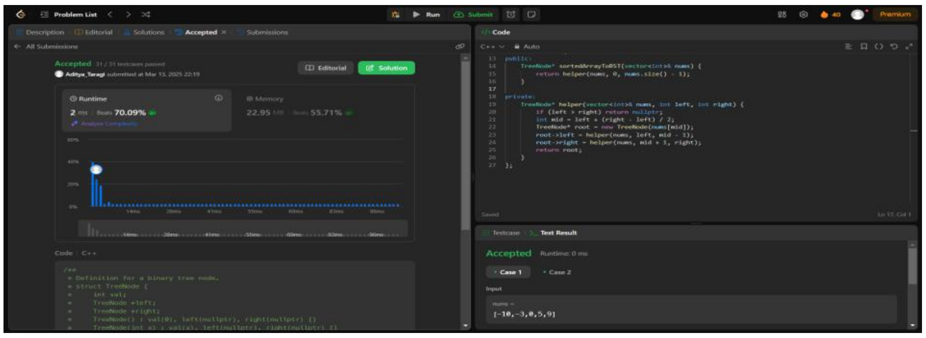
Subject Name: Advanced Programming Lab- 2

Problem 1: Convert Sorted Array to Binary Search Tree
(https://leetcode.com/problems/convert-sorted-array-to-binary-search-tree/ **)**

Code:

```
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;

  } };
```

Screenshot:



Accepted 31 / 31 testcases passed
Aditya_Taragi submitted at Mar 13, 2025 22:19

Runtime
2 ms | Beats 70.09%

Memory
22.95 MB | Beats 55.71%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
```

```
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;
    }
};
```

Accepted Runtime: 0 ms

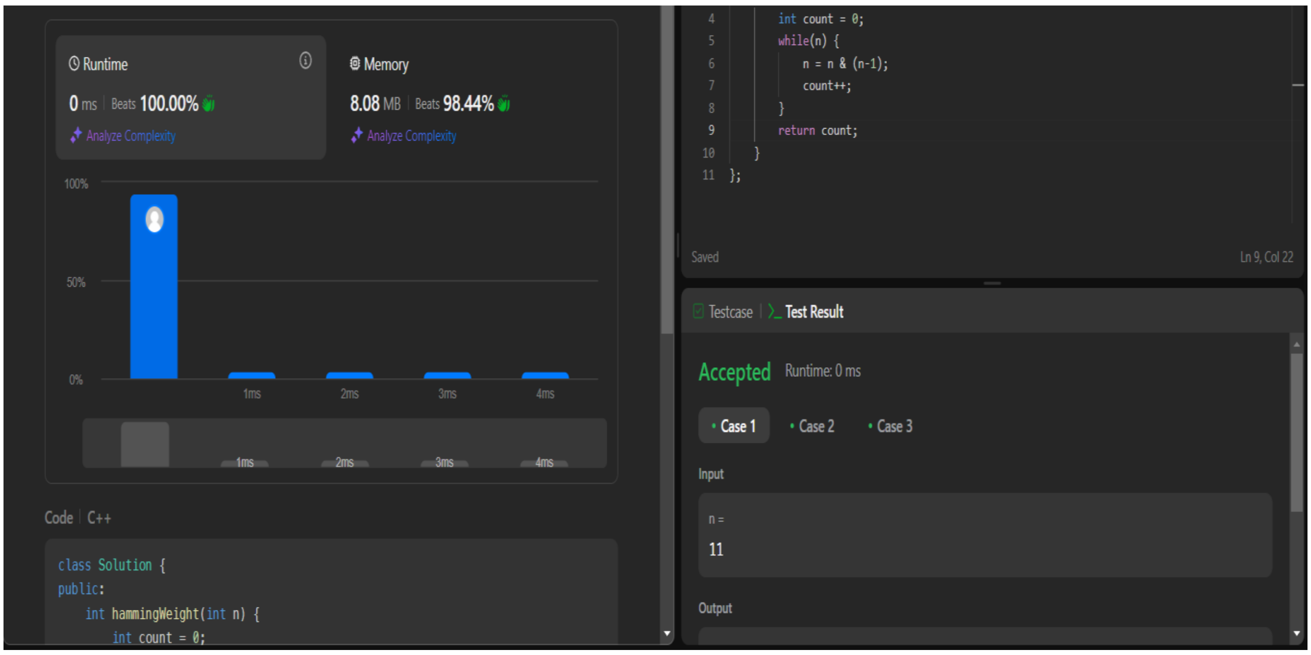
nums =
[-10,-3,0,5,9]

Problem 2: Number of 1 Bits (https://leetcode.com/problems/number-of-1-bits/)

Code:

```
class Solution {

public:

  int hammingWeight(int n) {

    int count = 0;

    while(n) {

      n = n & (n-1);

      count++;

    }

    return count;

  }

};
```

Screenshot:



Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

8.08 MB | Beats 98.44%

Analyze Complexity

Code | C++

```
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
```

```
        int count = 0;
        while(n) {
            n = n & (n-1);
            count++;
        }
        return count;
    }
};
```

Saved

Ln 9, Col 22

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

```
n =
11
```

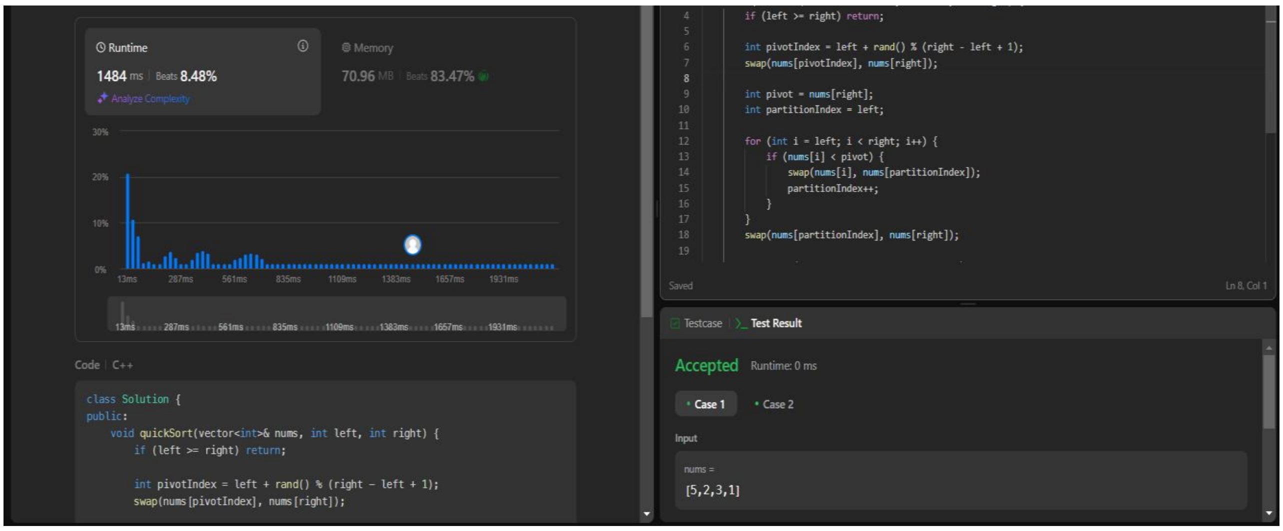
Output

Problem 3: Sort an Array (https://leetcode.com/problems/sort-an-array/)

Code:

```
class Solution {

public:

    void quickSort(vector<int>& nums, int left, int right) {

        if (left >= right) return;


        int pivotIndex = left + rand() % (right - left + 1);

        swap(nums[pivotIndex], nums[right]);


        int pivot = nums[right];

        int partitionIndex = left;


        for (int i = left; i < right; i++) {

            if (nums[i] < pivot) {

                swap(nums[i], nums[partitionIndex]);

                partitionIndex++;

            }

        }

        swap(nums[partitionIndex], nums[right]);


        quickSort(nums, left, partitionIndex - 1);

        quickSort(nums, partitionIndex + 1, right);

    }


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

        srand(time(0));

        quickSort(nums, 0, nums.size() - 1);
        return nums;
    }
};
```

Screenshot:



Runtime

1484 ms | Beats 8.48%

Analyze Complexity

Memory

70.96 MB | Beats 83.47%

Code | C++

```
class Solution {
public:
    void quickSort(vector<int>& nums, int left, int right) {
        if (left >= right) return;

        int pivotIndex = left + rand() % (right - left + 1);
        swap(nums[pivotIndex], nums[right]);
```

```
        if (left >= right) return;

        int pivotIndex = left + rand() % (right - left + 1);
        swap(nums[pivotIndex], nums[right]);

        int pivot = nums[right];
        int partitionIndex = left;

        for (int i = left; i < right; i++) {
            if (nums[i] < pivot) {
                swap(nums[i], nums[partitionIndex]);
                partitionIndex++;
            }
        }
        swap(nums[partitionIndex], nums[right]);

```

Saved

Ln 8, Col 1

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums =

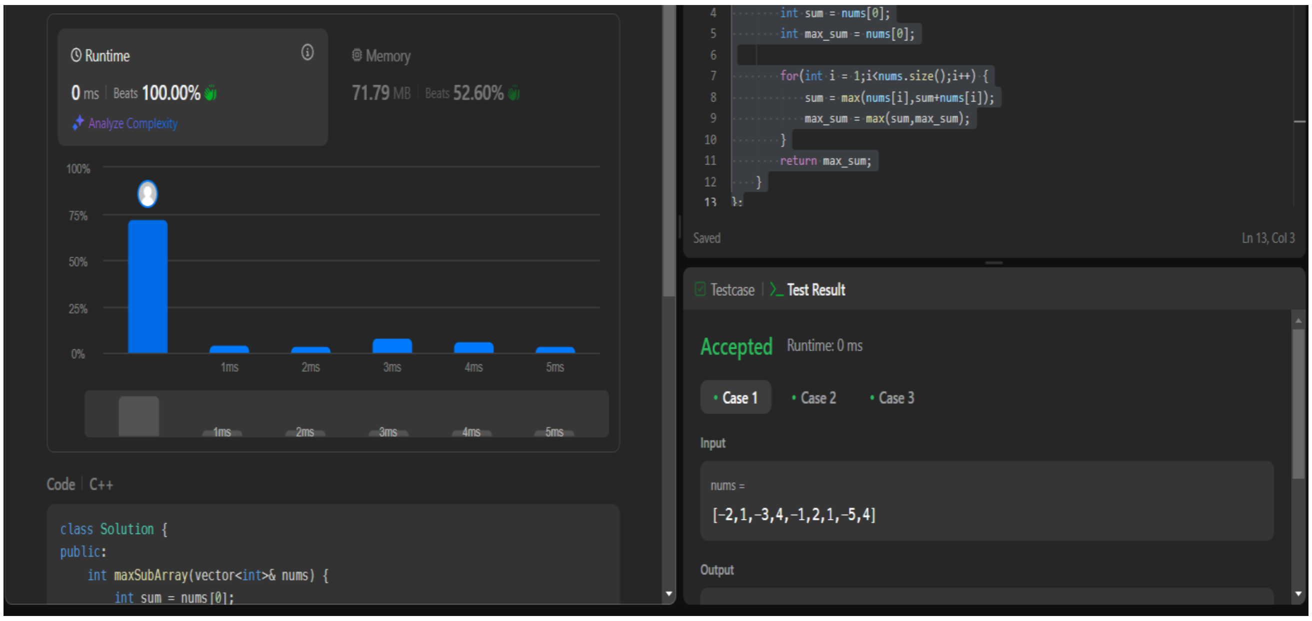
[5,2,3,1]

Problem 4: Maximum Subarray (https://leetcode.com/problems/maximum-subarray/)

Code:

```
class Solution {
public:
  void quickSort(vector<int>& nums, int left, int right) {
    if (left >= right) return;

    int pivot = nums[right];
    int partitionIndex = left;
    for (int i = left; i < right; i++) {
      if (nums[i] < pivot) {
        swap(nums[i], nums[partitionIndex]);
        partitionIndex++;
      }
    }
    swap(nums[partitionIndex], nums[right]);
    quickSort(nums, left, partitionIndex - 1);
    quickSort(nums, partitionIndex + 1, right);
  }
  vector<int> sortArray(vector<int>& nums) {
    quickSort(nums, 0, nums.size() - 1);
    return nums;
  }
};
```

Screenshot:



Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

71.79 MB | Beats 52.60%

Code | C++

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

```
        int sum = nums[0];
        int max_sum = nums[0];

        for(int i = 1;i<nums.size();i++) {
            sum = max(nums[i],sum+nums[i]);
            max_sum = max(sum,max_sum);
        }
        return max_sum;
    }
};
```

Saved   Ln 13, Col 3

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums =

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

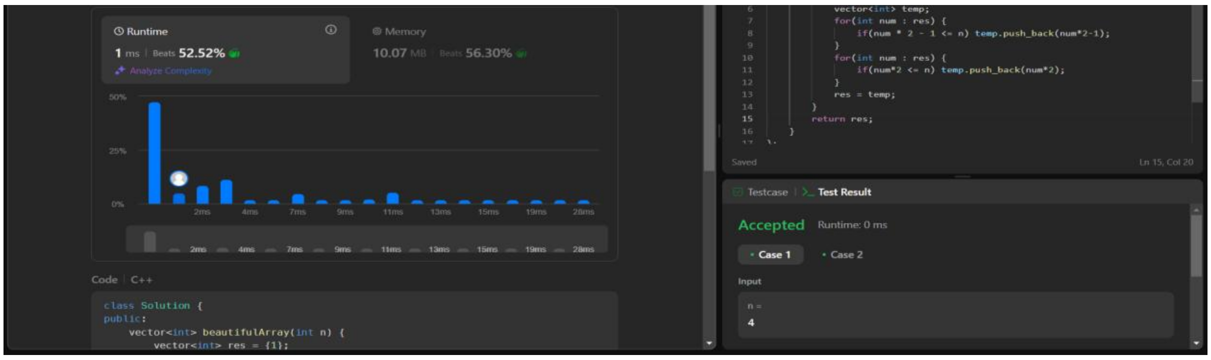
Output

Problem 5: Beautiful Array (https://leetcode.com/problems/beautiful-array/)

Code:

```
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;

  }

};
```

Screenshot:



Runtime
1 ms | Beats 52.52%
Analyze Complexity

Memory
10.07 MB | Beats 56.30%

Code | C++

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

```
            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;
    }
```

Saved   Ln 15, Col 20

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

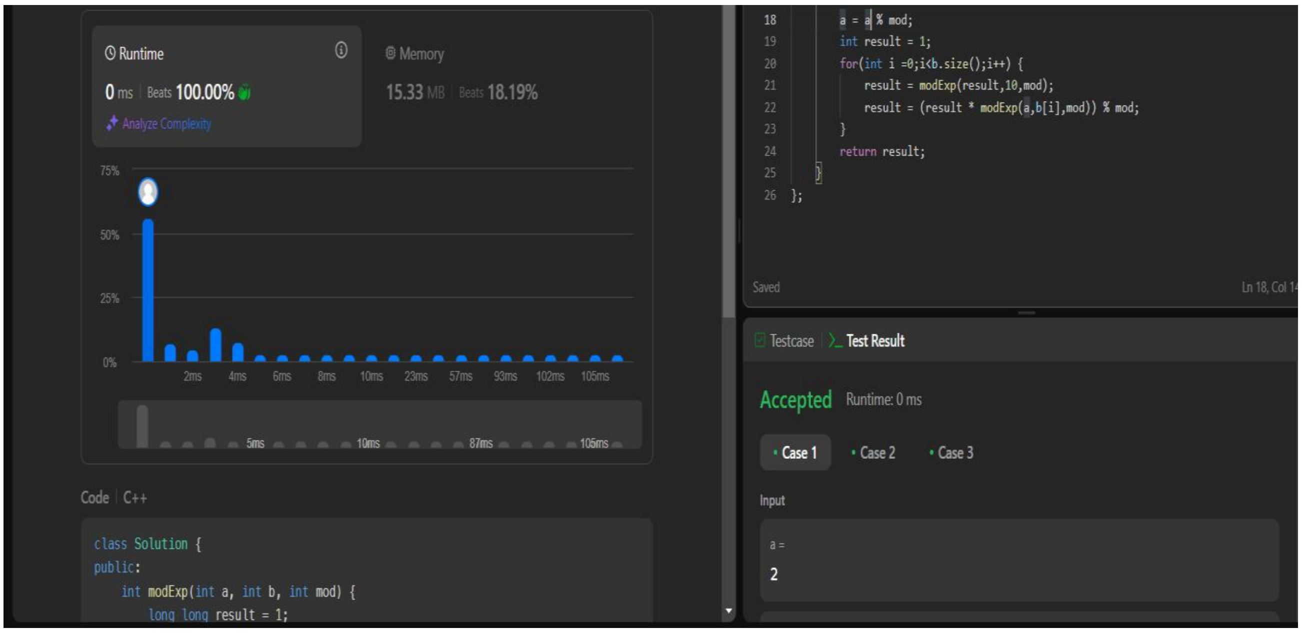
n =
4

Problem 6: Super Pow (https://leetcode.com/problems/super-pow/)

Code:

```
class Solution {
public:
  int modExp(int a, int b, int mod) {
    long long result = 1;
    a = a % mod;
    while(b>0) {
      if(b%2 == 1) {
        result = (result*a) % mod;
      }
      a = (a*a) % mod;
      b /=2;
    }
    return (int) result;
  }
  int superPow(int a, vector<int>& b) {
    int mod = 1337;
    a = a % mod;
    int result = 1;
    for(int i =0;i<b.size();i++) {
      result = modExp(result,10,mod);
      result = (result * modExp(a,b[i],mod)) % mod;
    }
    return result;
  }
};
```

Screenshot:



Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

15.33 MB | Beats 18.19%

Code | C++

```
class Solution {
public:
    int modExp(int a, int b, int mod) {
        long long result = 1;
```

```
        a = a % mod;
        int result = 1;
        for(int i =0;i<b.size();i++) {
            result = modExp(result,10,mod);
            result = (result * modExp(a,b[i],mod)) % mod;
        }
        return result;
    }
};
```

Saved   Ln 18, Col 1

Testcase | Test Result

Accepted   Runtime: 0 ms

Case 1   Case 2   Case 3

Input

a =

2

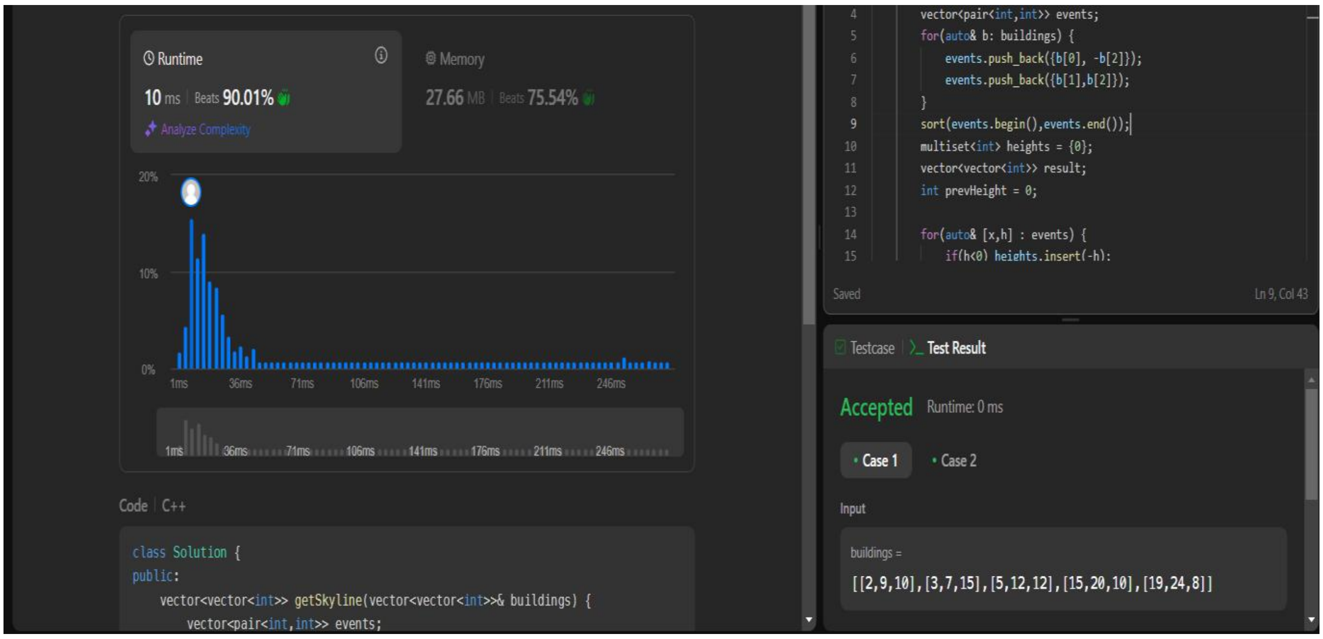
};

Problem 7: The Skyline Problem (https://leetcode.com/problems/the-skyline-problem/)

Code:

```
class Solution {

public:

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

    vector<pair<int,int>> events;

    for(auto& b: buildings) {

      events.push_back({b[0], -b[2]});

      events.push_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 maxHeight = *heights.rbegin();

      if(maxHeight != prevHeight) {

        result.push_back({x, maxHeight});

        prevHeight = maxHeight;

      }

    }

    return result;

  }

};
```

Screenshot:



```
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
        vector<pair<int,int>> events;
```

```
        vector<pair<int,int>> events;
        for(auto& b: buildings) {
            events.push_back({b[0], -b[2]});
            events.push_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);
```

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