



ASSIGNMENT 4

Name- Pranav kumar Singh

UID- 22BCS12894

Section-22BCS_IOT-606-B

1) Longest Nice Substring

class Solution {

public:

string longestNiceSubstring(string s) {

if (s.size() < 2) return "";

unordered_set<char> charSet(s.begin(), s.end());

for (int i = 0; i < s.size(); i++) {

if (charSet.count(tolower(s[i])) && charSet.count(toupper(s[i]))) continue;

string left = longestNiceSubstring(s.substr(0, i));

string right = longestNiceSubstring(s.substr(i + 1));

return left.size() >= right.size() ? left : right;

}

return s; };

The screenshot displays a C++ IDE with the following components:

- Problem List:** Shows the problem 'Longest Nice Substring' as 'Accepted'.
- Submissions:** Lists a submission by 'Rahul Kumar Pandey' submitted at 'Mar 17, 2025 10:01'.
- Runtime:** Shows '8 ms' and 'Beats 41.71%'.
- Memory:** Shows '14.17 MB' and 'Beats 63.14%'.
- Code Editor:** Contains the C++ code for the solution, which is a recursive function 'longestNiceSubstring'.
- Testcase:** Shows 'Case 1' as the selected test case.

```
1 class Solution {
2 public:
3     string longestNiceSubstring(string s) {
4         if (s.size() < 2) return "";
5
6         unordered_set<char> charSet(s.begin(), s.end());
7
8         for (int i = 0; i < s.size(); i++) {
9             char ch = s[i];
10            if (charSet.count(tolower(ch)) && charSet.count(toupper(ch))) {
11                continue;
12            }
13
14            string left = longestNiceSubstring(s.substr(0, i));
15            string right = longestNiceSubstring(s.substr(i + 1));
16
17            return left.size() >= right.size() ? left : right;
18        }
19        return s;
20    }
21 };
```



2) Reverse Bits

```
class Solution {
```

```
public:
```

```
    uint32_t reverseBits(uint32_t n) {
```

```
        uint32_t res = 0;
```

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

```
            res = (res << 1) | (n & 1);
```

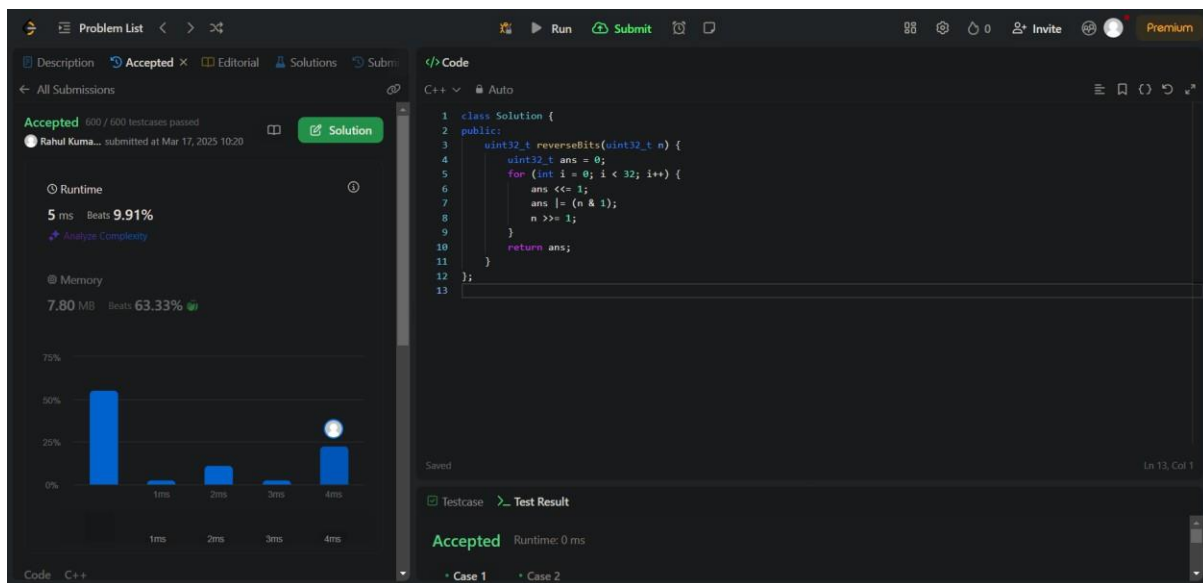
```
            n >>= 1;
```

```
        }
```

```
        return res;
```

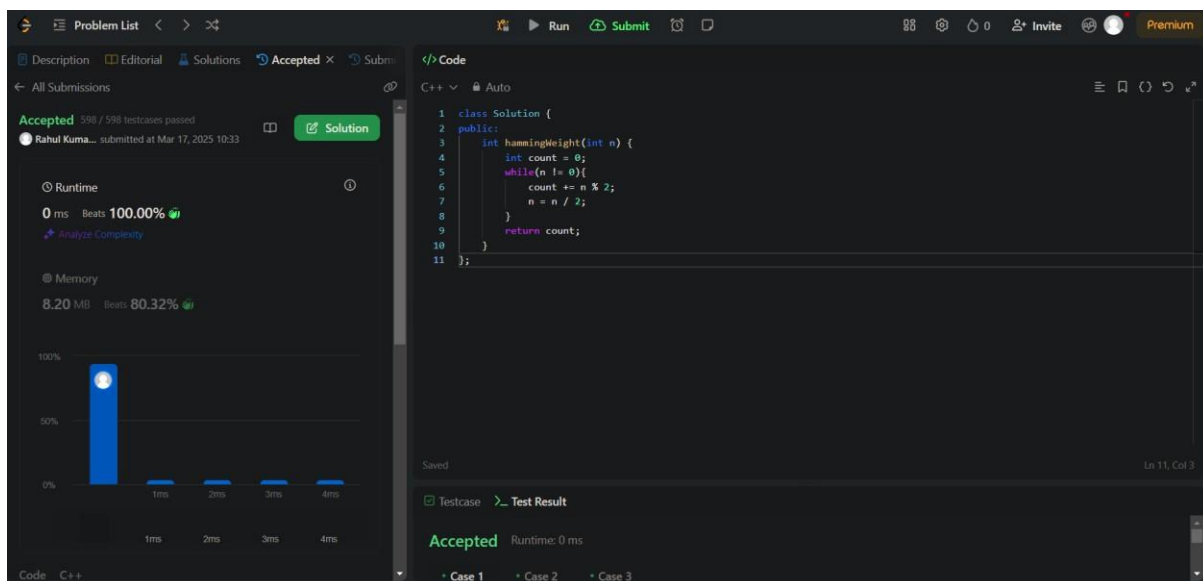
```
    }
```

```
};
```



3) Number of 1 Bits

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



4) Maximum Subarray

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int max_sum = nums[0], curr_sum = 0;
        for (int num : nums) {
            if (curr_sum < 0) curr_sum = 0;
            curr_sum += num;
            max_sum = max(max_sum, curr_sum);
        }
        return max_sum;
    }
};
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}  
};
```

Accepted 210 / 210 testcases passed
Rahul Kuma... submitted at Mar 17, 2025 10:35

Runtime
4 ms Beats 12.24%
[Analyze Complexity](#)

Memory
71.63 MB Beats 80.95%

Code

```
1 class Solution {  
2 public:  
3     int maxSubArray(vector<int>& arr) {  
4         int n=arr.size();  
5         long long maxi = LONG_MIN;  
6         long long sum = 0;  
7         for (int i = 0; i < n; i++) {  
8             sum += arr[i];  
9             if (sum > maxi) {  
10                 maxi = sum;  
11             }  
12             if (sum < 0) {  
13                 sum = 0;  
14             }  
15         }  
16         return maxi;  
17     }  
18 };
```

Testcase > Test Result

Case 1 Case 2 Case 3 +

</> Source



5) Search a 2D Matrix II

```
class Solution {  
public:  
    bool searchMatrix(vector<vector<int>>& matrix, int target) {  
        int row = 0, col = matrix[0].size() - 1;  
        while (row < matrix.size() && col >= 0) {  
            if (matrix[row][col] == target) return true;  
            else if (matrix[row][col] > target) col--;  
            else row++;  
        }  
        return false;  
    }  
};
```

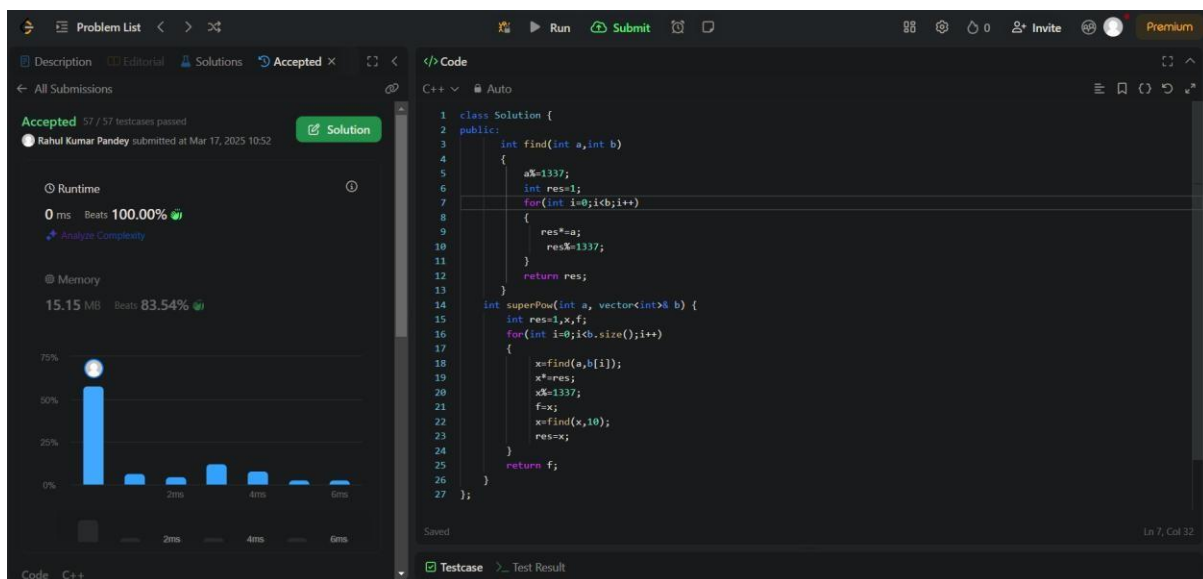
The screenshot displays a C++ IDE interface. On the left, the 'Problem List' tab is active, showing the problem 'Search a 2D Matrix II' as 'Accepted'. It indicates that 130 out of 130 testcases passed, submitted by 'Rahul Kuma...' on Mar 17, 2023 at 10:38. Below this, performance metrics are shown: Runtime is 57 ms (Beats 38.77%) and Memory is 18.73 MB (Beats 36.77%). A bar chart visualizes the runtime performance across different test cases. The main editor area shows the C++ code for the solution, which is a class 'Solution' with a public method 'searchMatrix'. The code uses a while loop to traverse the matrix from the top-right corner, adjusting the row and column pointers based on the comparison with the target. The bottom panel shows the 'Testcase' tab with 'Case 1' and 'Case 2' listed, and a 'Test Result' section.

6) Super Pow

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

    int power(int x, int y) {
        int res = 1;
        x %= MOD;
        while (y) {
            if (y % 2) res = (res * x) % MOD;
            x = (x * x) % MOD;
            y /= 2;
        }
        return res;
    }

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



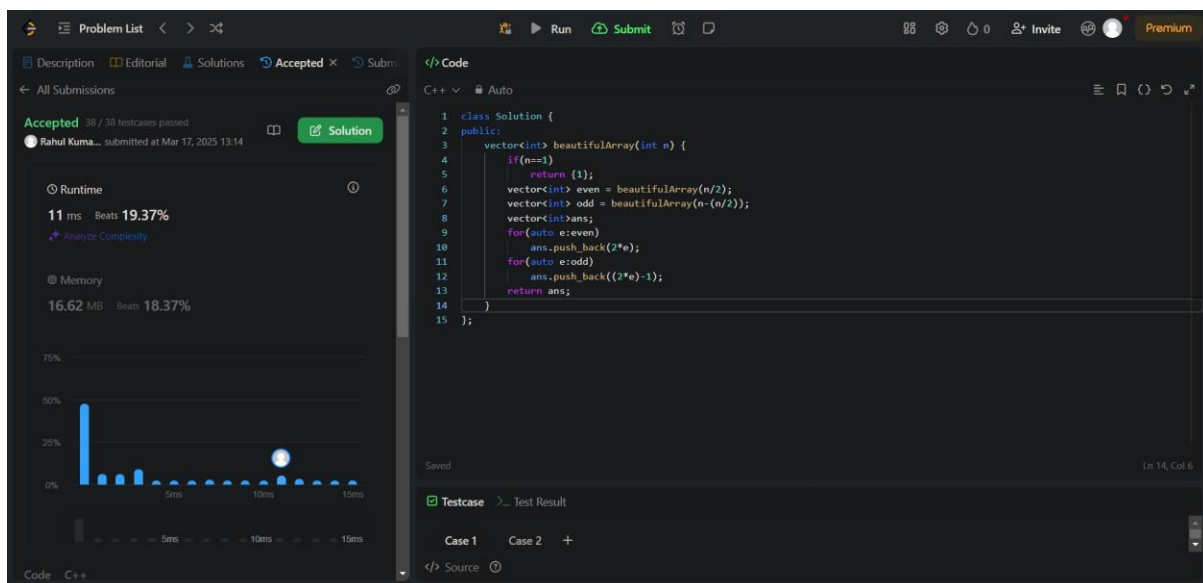
The screenshot displays a C++ code editor with the following code:

```
1 class Solution {
2 public:
3     int find(int a, int b)
4     {
5         a %= 1337;
6         int res = 1;
7         for (int i = 0; i < b; i++)
8         {
9             res = (res * a) % 1337;
10        }
11        return res;
12    }
13
14    int superPow(int a, vector<int>& b) {
15        int res = 1, x = a;
16        for (int i = 0; i < b.size(); i++)
17        {
18            x = find(x, b[i]);
19            res = (res * x) % 1337;
20            x = x % 1337;
21        }
22        return res;
23    }
24
25    int find(int a, int b)
26    {
27        a %= 1337;
28        int res = 1;
29        for (int i = 0; i < b; i++)
30        {
31            res = (res * a) % 1337;
32        }
33        return res;
34    }
35 }
```

The left panel shows the problem status as 'Accepted' with 57/57 testcases passed. The runtime is 0ms, and the memory is 15.15 MB. The right panel shows the C++ code implementing the solution.

7) Beautiful Array

```
class Solution {
public:
    vector<int> beautifulArray(int n) {
        vector<int> arr = {1};
        while (arr.size() < n) {
            vector<int> temp;
            for (int x : arr) if (x * 2 - 1 <= n) temp.push_back(x * 2 - 1);
            for (int x : arr) if (x * 2 <= n) temp.push_back(x * 2);
            arr = temp;
        }
        return arr;
    }
};
```





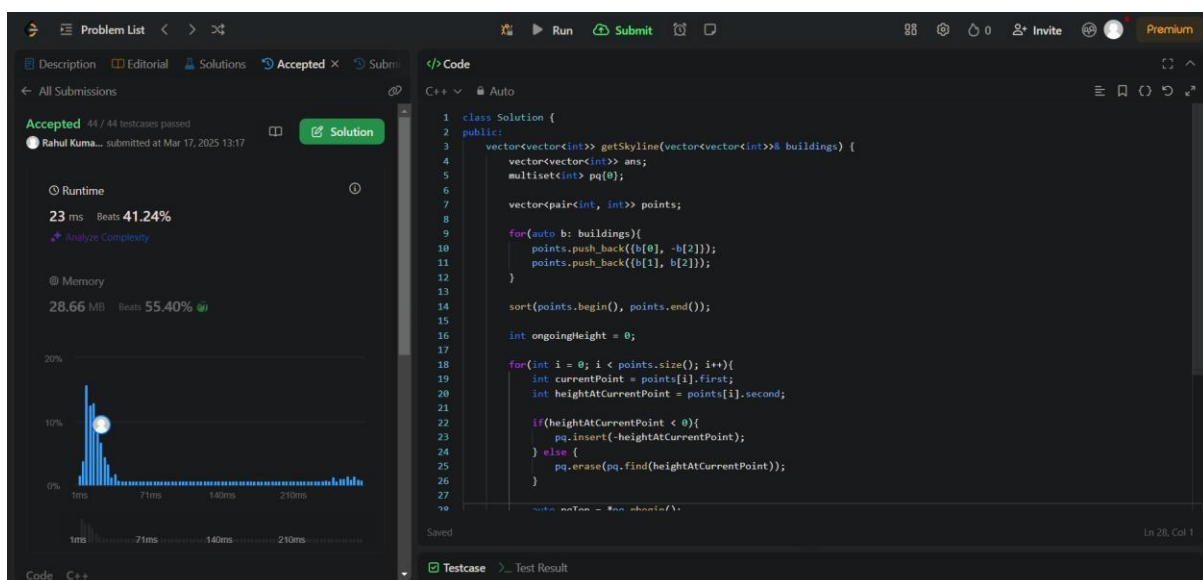
8) The Skyline_Problem

```
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>> res;
        int prev = 0;

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

            int curr = *heights.rbegin();
            if (curr != prev) {
                res.push_back({x, curr});
                prev = curr;
            }
        }
        return res;
    }
};
```

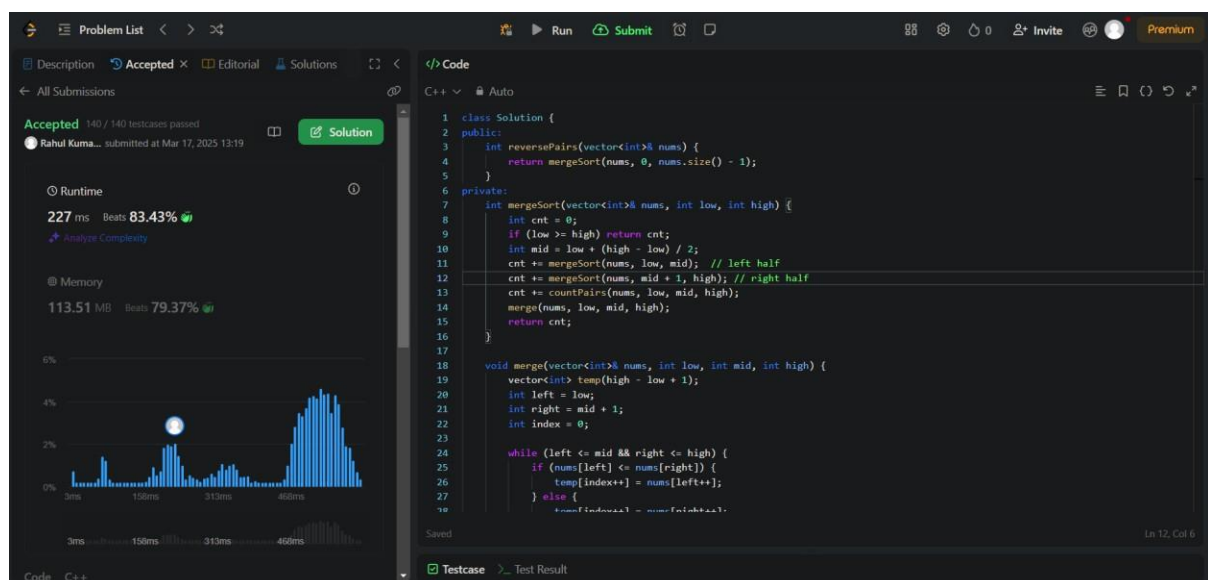


public:

```
int mergeAndCount(vector<int>& nums, int left, int mid, int right) {
    int count = 0, j = mid + 1;
    for (int i = left; i <= mid; i++) {
        while (j <= right && nums[i] > 2LL * nums[j]) j++;
        count += (j - (mid + 1));
    }
    sort(nums.begin() + left, nums.begin() + right + 1);
    return count;
}
```

```
int mergeSort(vector<int>& nums, int left, int right) {
    if (left >= right) return 0;
    int mid = (left + right) / 2;
    int count = mergeSort(nums, left, mid) + mergeSort(nums, mid + 1, right);
    count += mergeAndCount(nums, left, mid, right);
    return count;
}
```

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





10) Longest Increasing Subsequence II

```
class Solution {
public:
    vector<int> tree;
    void update(int node, int st, int end, int i, int val) {
        if (st == end) {
            tree[node] = max(tree[node], val);
            return;
        }
        int mid = (st + end) / 2;
        if (i <= mid) {
            update(node * 2, st, mid, i, val);
        } else {
            update(node * 2 + 1, mid + 1, end, i, val);
        }
        tree[node] = max(tree[node * 2], tree[node * 2 + 1]);
    }
    int query(int node, int st, int end, int x, int y) {
        if (x > end || y < st) return -1e9;
        if (st >= x && end <= y) {
            return tree[node];
        }
        int mid = (st + end) / 2;
        int left = query(2 * node, st, mid, x, y);
        int right = query(2 * node + 1, mid + 1, end, x, y);
        return max(left, right);
    }
    int lengthOfLIS(vector<int> & nums, int k) {
        int n = nums.size();
        if (n == 1) return 1;
        int m = *max_element(nums.begin(), nums.end());
        tree.clear();
        tree.resize(4 * m + 10);
        for (int i = n - 1; i >= 0; i--) {
            int l = nums[i] + 1, r = min(nums[i] + k, m);
            int x = query(1, 0, m, l, r);
            if (x == -1e9) x = 0;
            update(1, 0, m, nums[i], x + 1);
        }
        return tree[1];
    }
};
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}  
};
```

Problem List

Description | Accepted × | Editorial | Solutions

All Submissions

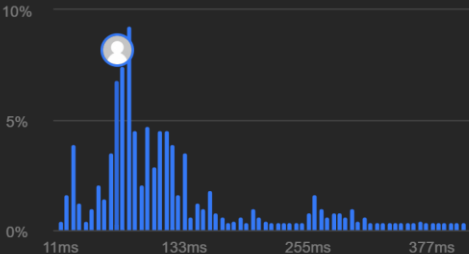
Accepted 84 / 84 testcases passed
Pranav Ku... submitted at Mar 17, 2025 22:21
Solution

Runtime

71 ms | Beats 75.78% 🌿
Analyze Complexity

Memory

59.88 MB | Beats 77.02% 🌿



Code

C++ Auto

```
1 class Solution {  
2 public:  
3     vector<int>tree;  
4     void update(int node,int st,int end,int i,int val){  
5         if(st==end){  
6             tree[node]=max(tree[node],val);  
7             return;  
8         }  
9         int mid=(st+end)/2;  
10        if(i<=mid){  
11            update(node*2,st,mid,i,val);  
12        }else{  
13            update(node*2+1,mid+1,end,i,val);  
14        }  
15    }  
16 };
```

Saved

Testcase | Test Result

Case 1 Case 2 Case 3 +

nums =
[4,2,1,4,3,4,5,8,15]

k =