

# ASSIGNMENT - 4

**Student Name: Manreet Kaur**

**UID: 22BCS15550**

**Branch: BE-CSE**

**Section/Group: 608/B**

**Semester: 6<sup>th</sup>**

**Subject Name: AP LAB**

## 1. Longest Nice Substring

The screenshot displays a coding platform interface. At the top, there's a navigation bar with a 'Problem List' tab selected. Below this, a table shows the submission history for the 'Longest Nice Substring' problem. The table has columns for Status, Language, Runtime, Memory, and Notes. Three submissions are listed, all with a status of 'Accepted'. The first submission (ID 1) was made on Feb 27, 2025, in C++ with a runtime of 366 ms and memory of 122.6 MB. The second submission (ID 2) was made on Feb 28, 2025, in C++ with a runtime of 4 ms and memory of 11.5 MB. The third submission (ID 3) was made on Mar 06, 2025, in C++ with a runtime of 361 ms and memory of 122.6 MB.

Below the submission history, the 'Code' editor is open, showing the C++ code for the 'Longest Nice Substring' problem. The code defines a class 'Solution' with a public method 'longestNiceSubstring' that takes a string 's' and returns the longest nice substring. The code uses a sliding window approach to find the longest substring where the number of lowercase and uppercase letters is equal.

```
1 class Solution {
2 public:
3     string longestNiceSubstring(string s) {
4         int n=s.length();
5         if (s.length()<2) {
6             return "";
7         }
8         bool lower[26]={false};
9         bool upper[26]={false};
10        for(char c:s){
11            if(islower(c)){
12                lower[c-'a']=true;
13            }
14            else{
15                upper[c-'A']=true;
16            }
17        }
18        for(int i=0;i<n;i++){
19            char c=s[i];
20            if(islower(c)&&!upper[c-'a']){
21                string left=longestNiceSubstring(s.substr(0,i));
22                string right=longestNiceSubstring(s.substr(i+1));
23                return left.length()>=right.length()?left:right;
```

On the right side of the code editor, there's a user profile for 'Manreet\_05' with a premium subscription. Below the profile, there are buttons for 'My Lists', 'Notebook', 'Submissions', 'Progress', and 'Points'. At the bottom, there are links for 'Try New Features', 'Orders', 'My Playgrounds', 'Settings', and 'Classic Mode'.

## 2. Reverse Bits

The screenshot shows a submission page for the 'Reverse Bits' problem. At the top, there's a navigation bar with 'Problem List', 'Description', 'Editorial', 'Solutions', and 'Submissions'. Below this is a table of submissions. The first submission (ID 1) is 'Accepted', made 31 minutes ago, using C++ with a runtime of 0 ms and memory of 7.7 MB. Below the table is a code editor showing the following C++ code:

```
1 class Solution {
2 public:
3     uint32_t reverseBits(uint32_t n) {
4         uint32_t result = 0;
5         for (int i = 0; i < 32; ++i) {
6             result = (result << 1) | (n & 1);
7             n >>= 1;
8         }
9         return result;
10    }
11 };
```

On the right side, there's a user profile for 'Manreet\_05' with a prompt to 'Access all features with our Premium subscription!'. Below the profile are icons for 'My Lists', 'Notebook', 'Submissions', 'Progress', and 'Points'.

## 3. Number of 1 Bits


The screenshot shows a list of submissions for the 'Number of 1 Bits' problem. The navigation bar includes 'Problem List', 'Description', 'Accepted' (with a close icon), 'Editorial', 'Solutions', and 'Submissions'. The submission table shows two entries:


	Status	Language	Runtime	Memory	Notes
2	Accepted 27 minutes ago	C++	0 ms	8.2 MB	
1	Accepted 30 minutes ago	C++	0 ms	8.2 MB	


</> Code


C++ ▾ 🔒 Auto


```
1 class Solution {
2 public:
3     int hammingWeight(uint32_t n) {
4         int count = 0;
5         while (n) {
6             n &= (n - 1);
7             count++;
8         }
9         return count;
10    }
11 };
```


 **Manreet\_05**  
Access all features with our Premium subscription!

 My Lists






 Notebook


 Submissions

 Progress

 Points

#### 4. Maximum Subarray

  Problem List   




Description | Accepted × | Editorial | Solutions | Submissions


	Status ▾	Language ▾	Runtime	Memory	Notes
1	Accepted 26 minutes ago	C++	0 ms	71.8 MB	


</> Code


C++ ▾ 🔒 Auto


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


 **Manreet\_05**  
Access all features with our Premium subscription!


 My Lists


 Notebook

 Submissions

 Progress

 Points

 Try New Features

 Orders

## 5. Search a 2D Matrix II

The screenshot shows a code editor interface with a dark theme. At the top, there's a navigation bar with 'Problem List', '<', '>', and a refresh icon. Below it, tabs for 'Description', 'Accepted', 'Editorial', 'Solutions', and 'Submissions' are visible. A table below the tabs shows submission details: '1', 'Accepted', '25 minutes ago', 'C++', '51 ms', and '18.6 MB'. The main area is a code editor with a C++ file named 'Solution.cpp'. The code implements a search function for a 2D matrix. On the right, a sidebar shows the user profile 'Manreet\_05' with a premium subscription notice and icons for 'My Lists', 'Notebook', 'Submissions', 'Progress', and 'Points'.

```
1 class Solution {
2 public:
3     bool searchMatrix(vector<vector<int>>& matrix, int target)
4     {
5         if (matrix.empty() || matrix[0].empty()) return false;
6
7         int rows = matrix.size();
8         int cols = matrix[0].size();
9         int row = 0;
10        int col = cols - 1;
11
12        while (row < rows && col >= 0) {
13            if (matrix[row][col] == target) {
14                return true;
15            } else if (matrix[row][col] > target) {
16                col--;
17            } else {
18                row++;
19            }
20        }
21        return false;
22    }
23 };
```

## 6. Super Pow

The screenshot shows a code editor interface with a dark theme. At the top, there's a navigation bar with 'Problem List', '<', '>', and a refresh icon. Below it, tabs for 'Description', 'Editorial', 'Solutions', and 'Submissions' are visible. A table below the tabs shows submission details: '1', 'Accepted', 'Mar 13, 2025', 'C++', '1 ms', and '15.1 MB'.



<div> <div> <div>Problem List</div> <div>&lt; &gt; ↺</div> </div> <div> </div> </div>				
<div> <div>Description</div> <div>Editorial</div> <div>Solutions</div> <div>Submissions</div> </div>				
<div> <div>Status ▾</div> <div>Language ▾</div> <div>Runtime</div> <div>Memory</div> <div>Notes</div> </div>				
1	<div>Accepted</div> <div>12 minutes ago</div>	<div>C++</div>	<div>0 ms</div>	<div>10.1 MB</div>

## 8. The Skyline Problem

Problem List

< > ↺

Description

Accepted

Editorial

Solutions

Submissions

Status ▾

Language ▾

Runtime

Memory

Notes

1

Accepted

20 minutes ago

C++

159 ms

105.3 MB

Code

C++ ▾

Auto

```

1 class Solution {
2 public:
3     vector<vector<int>> mergeskylines(vector<vector<int>> merged;
4     vector<vector<int>> merged;
5     int h1 = 0, h2 = 0;
6     int i = 0, j = 0;
7
8     while (i < left.size() && j < right.size()) {
9         int x, h;
10        if (left[i][0] < right[j][0]) {
11            x = left[i][0];
12            h1 = left[i][1];
13            h = max(h1, h2);
14            i++;
15        } else if (left[i][0] > right[j][0]) {
16            x = right[j][0];
17            h2 = right[j][1];
18            h = max(h1, h2);
19            j++;
20        } else {
21            x = left[i][0];
22            h1 = left[i][1];
23            h2 = right[j][1];

```

Manreet\_05

Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features

Orders

My Playgrounds

Settings

Classic Mode

## 9. Reverse Pairs

The screenshot shows a code editor interface for a problem titled "Reverse Pairs". The top navigation bar includes "Problem List", "Description", "Accepted", "Editorial", "Solutions", and "Submissions". Below this is a table with columns: Status, Language, Runtime, Memory, and Notes. The first row shows a submission with status "Accepted", language "C++", runtime "545 ms", and memory "243.5 MB".

The code editor displays the following C++ code:

```
1 class Solution {
2 public:
3
4 int mergeAndCount(vector<int>& nums, int left, int mi
5     int count = 0;
6     int j = mid + 1;
7
8     for (int i = left; i <= mid; i++) {
9         while (j <= right && nums[i] > 2LL * nums[j])
10             j++;
11     }
12     count += (j - (mid + 1));
13 }
14
15 vector<int> temp;
16 int i = left, k = mid + 1;
17
18 while (i <= mid && k <= right) {
19     if (nums[i] <= nums[k]) {
20         temp.push_back(nums[i++]);
21     } else {
22         temp.push_back(nums[k++]);
23     }
```

On the right side of the editor, there is a user profile for "Manreet\_05" with a premium subscription. Below the profile are buttons for "My Lists", "Notebook", "Submissions", "Progress", and "Points". At the bottom, there are links for "Try New Features", "Orders", "My Playgrounds", "Settings", and "Classic Mode".

## 10. Longest Increasing Subsequence II

The screenshot shows a code editor interface for a problem titled "Longest Increasing Subsequence II". The top navigation bar includes "Problem List", "Description", "Editorial", "Solutions", and "Submissions". Below this is a table with columns: Status, Language, Runtime, Memory, and Notes. The first row shows a submission with status "Accepted", language "C++", runtime "26 ms", and memory "54.5 MB".

The code editor displays the following C++ code:

```
1 class Solution {
2 public:
3
4 int longestIncreasingSubsequence(vector<int>& nums) {
5     int n = nums.size();
6     vector<int> dp(n, 1);
7
8     for (int i = 1; i < n; i++) {
9         for (int j = 0; j < i; j++) {
10             if (nums[i] > nums[j]) {
11                 dp[i] = max(dp[i], dp[j] + 1);
12             }
13         }
14     }
15     return *max_element(dp.begin(), dp.end());
16 }
```

On the right side of the editor, there is a user profile for "Manreet\_05" with a premium subscription. Below the profile are buttons for "My Lists", "Notebook", "Submissions", "Progress", and "Points". At the bottom, there are links for "Try New Features", "Orders", "My Playgrounds", "Settings", and "Classic Mode".

 Code

C++   Auto

```
1 class SegmentTree {
2 public:
3     vector<int> tree;
4     int size;
5
6     SegmentTree(int n) {
7         size = n;
8         tree.resize(2 * n, 0);
9     }
10
11    void update(int index, int value) {
12        index += size;
13        tree[index] = max(tree[index], value);
14        while (index > 1) {
15            index /= 2;
16            tree[index] = max(tree[2 * index], tree[2 * index + 1]);
17        }
18    }
19    int query(int left, int right) {
20        int result = 0;
21        left += size;
22        right += size;
23        while (left < right) {
```



**Manreet\_05**

Access all features with our  
Premium subscription!



My Lists



Notebook



Submissions



Progress



Points



Try New Features



Orders



My Playgrounds



Settings



Classic Mode