

ASSIGNMENT

Student Name: Sandhya Bharti

UID: 22BCS11412

Branch: BE-CSE

Section/Group: 608/B

Semester: 6th

Subject Name: AP LAB

1. Longest Nice Substring:

	Description	Editorial	Solutions	Submissions	
	Status ▾	Language ▾	Runtime	Memory	Notes
2	Accepted Mar 06, 2025	C++	369 ms	122.5 MB	
1	Accepted Feb 28, 2025	C++	19 ms	14 MB	

```
Code
C++ ▾ 🔒 Auto
1 class Solution {
2 public:
3 bool isNice(const string& s) {
4 unordered_set<char> charSet(s.begin(), s.end()); // Store
5 for (char c : s) {
6 if (isalpha(c)) {
7 if (charSet.find(tolower(c)) == charSet.end() || c
8 return false;
9 }
10 }
11 }
12 return true;
13 }
14
15 string longestNiceSubstring(string s) {
16 int n = s.length();
17 string longestNice = "";
18
19 for (int i = 0; i < n; i++) {
20 for (int j = i; j < n; j++) {
21 string sub = s.substr(i, j - i + 1);
22 if (isNice(sub) && sub.length() > longestNice.leng
23 longestNice = sub;
24 }
25 }
26 }
27 return longestNice;
28 }
29 }
```

sandhya
Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features •

Orders

My Playgrounds

Settings

Classic Mode

Appearance >

Sign Out

2. Reverse Bits:

Description Accepted × Editorial Solutions Submissions				
Status ▾	Language ▾	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	0 ms	7.8 MB	

```
Code
C++ ▾ Auto
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 <<= 1;
7             result |= (n & 1);
8             n >>= 1;
9         }
10        return result;
11    }
12 };
13
```

S sandhya
Access all features with our
Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features •

3. Number of 1 Bits:

Description Accepted × Editorial Solutions Submissions				
Status ▾	Language ▾	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	0 ms	8.2 MB	

Code

C++ Auto

```
1 class Solution {
2 public:
3     int hammingWeight(int n) {
4         int count = 0;
5         while (n) {
6             count += (n & 1); // Add 1 if the last bit is set
7             n >>= 1; // Shift right to process the next bit
8         }
9         return count;
10    }
11 };
```

S sandhya

Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

4. Maximum Subarray:

Description Editorial Solutions Submissions					
Status	Language	Runtime	Memory	Notes	
1 Accepted 11 minutes ago	C++	1 ms	312.2 MB		

Code

C++ Auto

```
1 #include <vector>
2 using namespace std;
3
4 class Solution {
5 public:
6     int maxSubArray(vector<int>& nums) {
7         int maxSum = nums[0];
8         int currentSum = nums[0];
9
10        for (int i = 1; i < nums.size(); i++) {
11            currentSum = max(nums[i], currentSum + nums[i]);
12            maxSum = max(maxSum, currentSum);
13        }
14
15        return maxSum;
16    }
17 };
18
```

S sandhya

Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features

Orders

My Playgrounds

5. Search a 2D Matrix II:

Description Accepted × Editorial Solutions Submissions				
Status ▾	Language ▾	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	54 ms	18.9 MB	

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

S sandhya
Access all features with our
Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features •

Orders

My Playgrounds

Settings

6. Super Pow:

Description Accepted × Editorial Solutions Submissions				
Status ▾	Language ▾	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	8 ms	11.9 MB	

```
Code
C++ v Auto
1 class Solution {
2 public:
3     const int MOD = 1337;
4
5     int modPow(int a, int b) {
6         int result = 1;
7         a %= MOD;
8         while (b > 0) {
9             if (b % 2 == 1) {
10                 result = (result * a) % MOD;
11             }
12             a = (a * a) % MOD;
13             b /= 2;
14         }
15         return result;
16     }
17
18     int superPow(int a, vector<int>& b) {
19         int result = 1;
20         for (int digit : b) {
21             result = modPow(result, 10) * modPow(a, digit) % MOD;
22         }
23     }
24 }
```

Sandhya
Access all features with our Premium subscription!

My Lists Notebook Submissions

Progress Points

Try New Features •

Orders

My Playgrounds

Settings

Classic Mode

Ln 26, Col 1

7. Beautiful Array:

Status	Language	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	0 ms	16.5 MB	

```
Code
C++ v Auto
1 class Solution {
2 public:
3     vector<int> beautifulArray(int n) {
4         vector<int> result = {1};
5         while (result.size() < n) {
6             vector<int> temp;
7             for (int num : result) {
8                 if (2 * num - 1 <= n) temp.push_back(2 * num - 1);
9             }
10            for (int num : result) {
11                if (2 * num <= n) temp.push_back(2 * num);
12            }
13            result = temp;
14        }
15        return result;
16    }
17 };
18 }
```

Sandhya
Access all features with our Premium subscription!

My Lists Notebook Submissions

Progress Points

Try New Features •

Orders

My Playgrounds

8. The Skyline Problem:

The screenshot shows a submission interface for a C++ program. The submission is labeled '1' and is 'Accepted' in 3 minutes. The language is C++, the runtime is 18 ms, and the memory is 57.2 MB. The code is as follows:

```
1 class Solution {
2 public:
3     vector<vector<int>> getSkyline(vector<vector<int>>& buildings)
4     {
5         vector<pair<int, int>> events;
6         for (auto& b : buildings) {
7             events.emplace_back(b[0], -b[2]); // Start of building
8             events.emplace_back(b[1], b[2]); // End of building
9         }
10        sort(events.begin(), events.end());
11
12        multiset<int> heights = {0};
13        vector<vector<int>> result;
14        int prevHeight = 0;
15
16        for (auto& event : events) {
17            int x = event.first, h = event.second;
18            if (h < 0) {
19                heights.insert(-h); // Add building height
20            } else {
21                heights.erase(heights.find(h)); // Remove building height
22            }
23            int currHeight = *heights.rbegin();
24            if (currHeight != prevHeight) {
25                result.push_back({x, currHeight});
26                prevHeight = currHeight;
27            }
28        }
29        return result;
30    }
31 }
```

On the right side of the interface, there is a sidebar for the user 'sandhya'. It includes a profile section with a message 'Access all features with our Premium subscription!' and several icons for 'My Lists', 'Notebook', 'Submissions', 'Progress', and 'Points'. Below these are links for 'Try New Features', 'Orders', 'My Playgrounds', 'Settings', 'Classic Mode', 'Appearance', and 'Sign Out'.

9. Reverse Pairs:

The screenshot shows a submission interface for a C++ program. The submission is labeled '1' and is 'Accepted' in 3 minutes. The language is C++, the runtime is 5 ms, and the memory is 18.3 MB. The submission details are as follows:

Status	Language	Runtime	Memory	Notes
1 Accepted in 3 minutes	C++	5 ms	18.3 MB	

Code

C++ Auto

```
1 class Solution {
2 public:
3     int mergeAndCount(vector<int>& nums, int left, int mid, int right) {
4         int count = 0, j = mid + 1;
5         for (int i = left; i <= mid; i++) {
6             while (j <= right && nums[i] > 2LL * nums[j]) j++;
7             count += (j - (mid + 1));
8         }
9
10        vector<int> temp;
11        int i = left, k = mid + 1;
12        while (i <= mid && k <= right) {
13            if (nums[i] <= nums[k]) temp.push_back(nums[i++]);
14            else temp.push_back(nums[k++]);
15        }
16        while (i <= mid) temp.push_back(nums[i++]);
17        while (k <= right) temp.push_back(nums[k++]);
18
19        for (int i = left; i <= right; i++) nums[i] = temp[i - left];
20        return count;
21    }
22
23    int mergeSort(vector<int>& nums, int left, int right) {
24        if (left >= right) return 0;
25        int mid = left + (right - left) / 2;
26        int count = mergeSort(nums, left, mid) + mergeSort(nums, mid + 1, right);
27        count += mergeAndCount(nums, left, mid, right);
28        return count;
29    }
30 }
```

sandhya

Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features

Orders

My Playgrounds

Settings

Classic Mode

Appearance

Sign Out

10. Longest Increasing Subsequence II:

Code

C++ Auto

```
1 class Solution {
2 public:
3     int lengthOfLIS(vector<int>& nums, int k) {
4         map<int, int> dp;
5         int maxLength = 0;
6
7         for (int num : nums) {
8             int best = 0;
9             for (int i = num - k; i <= num - 1; i++) {
10                 if (dp.count(i)) {
11                     best = max(best, dp[i]);
12                 }
13             }
14             dp[num] = best + 1;
15             maxLength = max(maxLength, dp[num]);
16         }
17
18         return maxLength;
19     }
20 };
```

sandhya

Access all features with our Premium subscription!

My Lists

Notebook

Submissions

Progress

Points

Try New Features

Orders

My Playgrounds

Settings