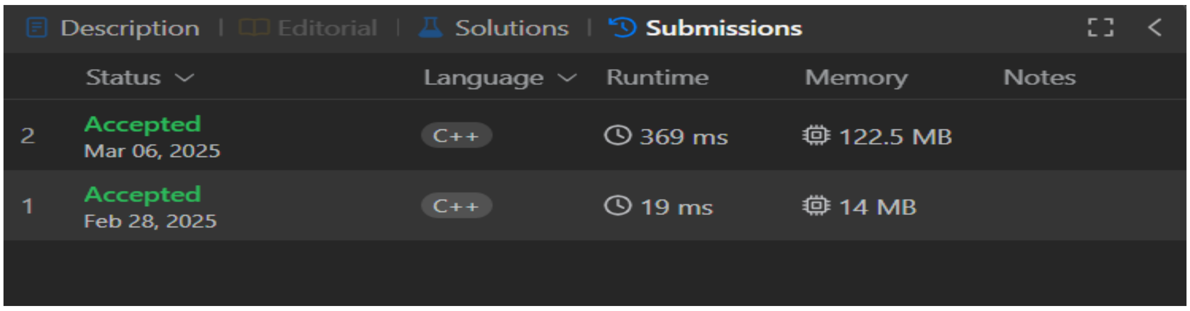
ASSIGNMENT

Student Name: Sandhya Bharti **UID: 22BCS11412**

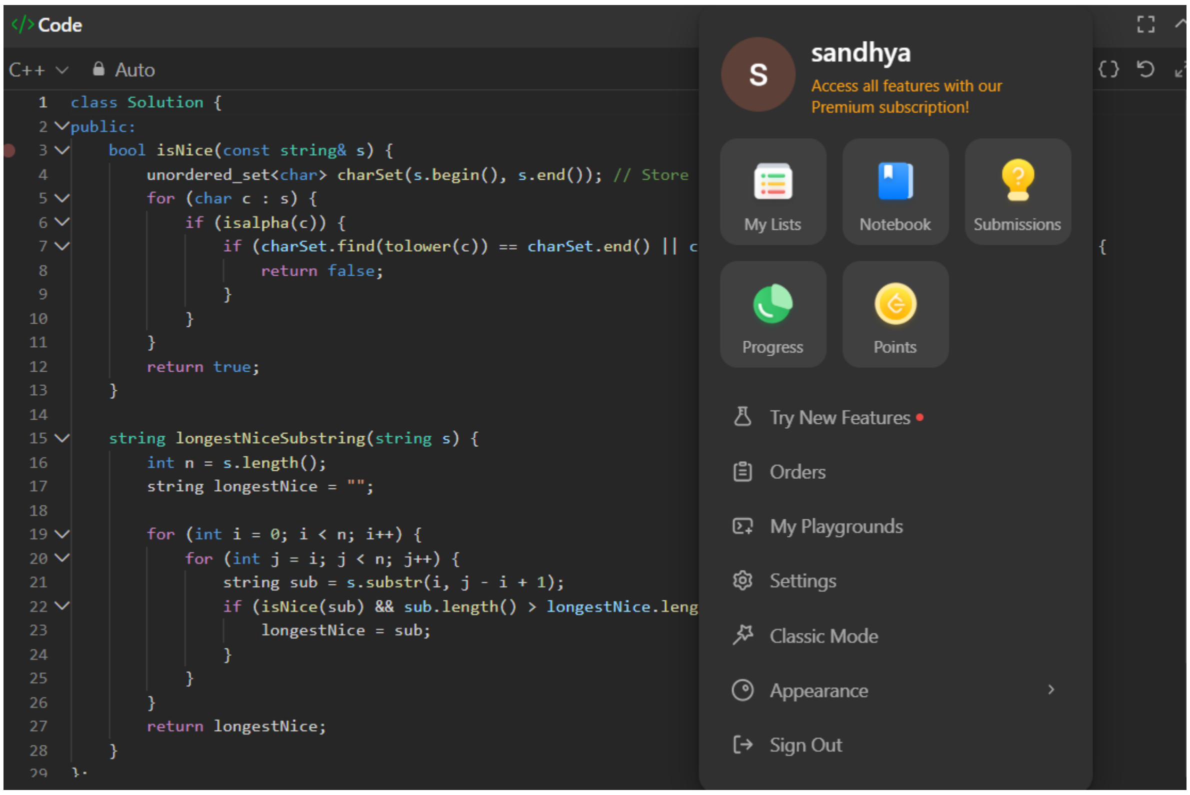
Branch: BE-CSE **Section/Group: 608/B**

Semester: 6th **Subject Name: AP LAB**

1. Longest Nice Substring:

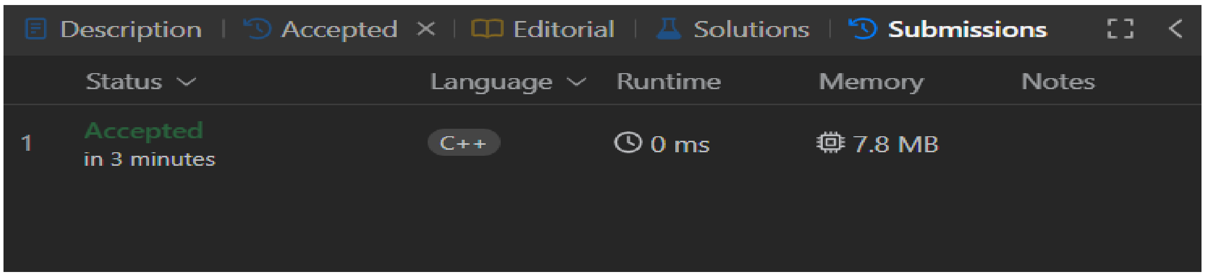


	Status	Language	Runtime	Memory	Notes
2	Accepted Mar 06, 2025	C++	369 ms	122.5 MB	
1	Accepted Feb 28, 2025	C++	19 ms	14 MB	

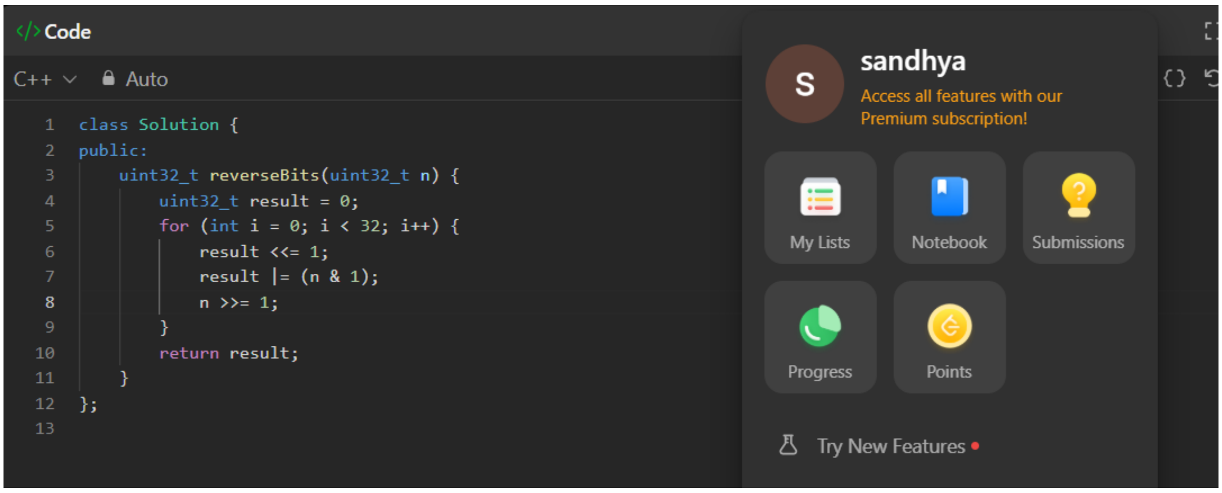


```cpp
class Solution {
public:
    bool isNice(const string& s) {
        unordered_set<char> charSet(s.begin(), s.end()); // Store
        for (char c : s) {
            if (isalpha(c)) {
                if (charSet.find(tolower(c)) == charSet.end() || c
                    return false;
                }
            }
        }
        return true;
    }

    string longestNiceSubstring(string s) {
        int n = s.length();
        string longestNice = "";

        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                string sub = s.substr(i, j - i + 1);
                if (isNice(sub) && sub.length() > longestNice.leng
                    longestNice = sub;
                }
            }
        }
        return longestNice;
    }
};
```

2. Reverse Bits:

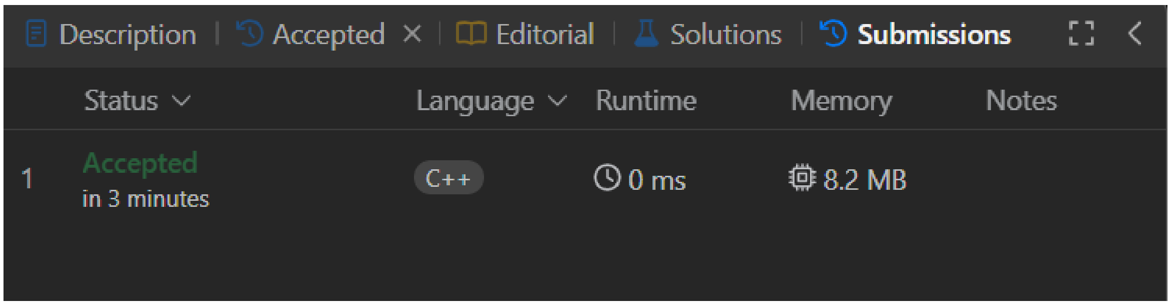


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	0 ms	7.8 MB	

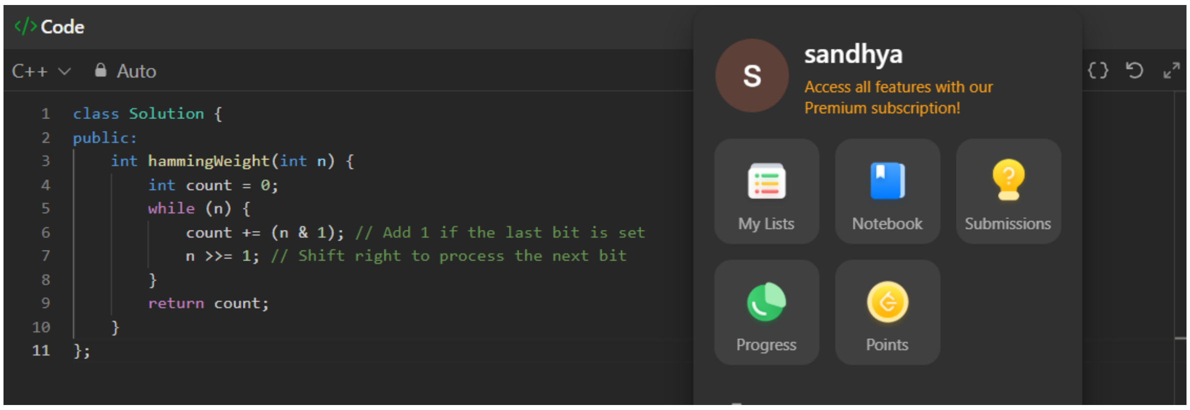


```
class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t result = 0;
        for (int i = 0; i < 32; i++) {
            result <<= 1;
            result |= (n & 1);
            n >>= 1;
        }
        return result;
    }
};
```

3. Number of 1 Bits:

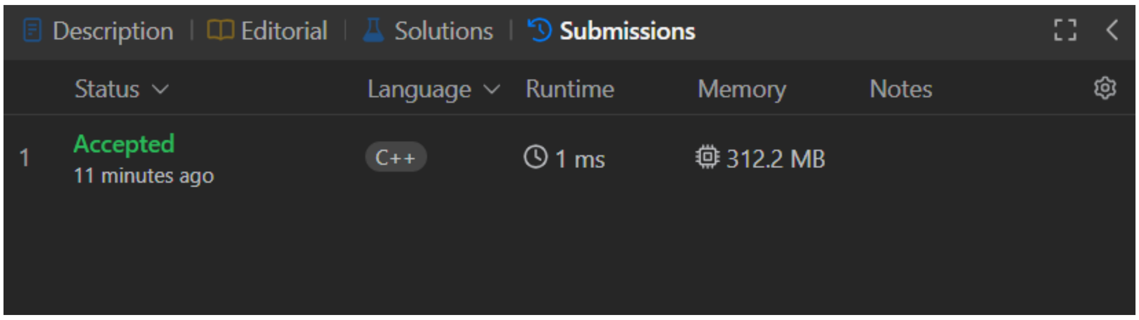


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	0 ms	8.2 MB	

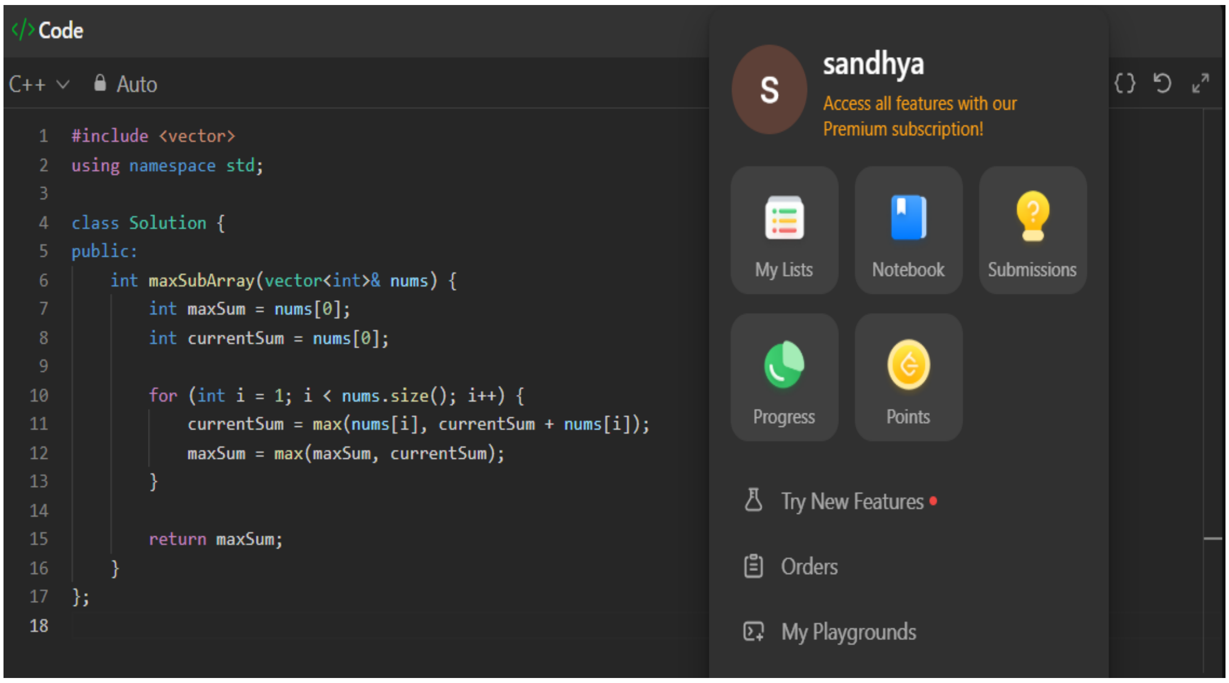


```cpp
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
        while (n) {
            count += (n & 1); // Add 1 if the last bit is set
            n >>= 1; // Shift right to process the next bit
        }
        return count;
    }
};
```

4. Maximum Subarray:

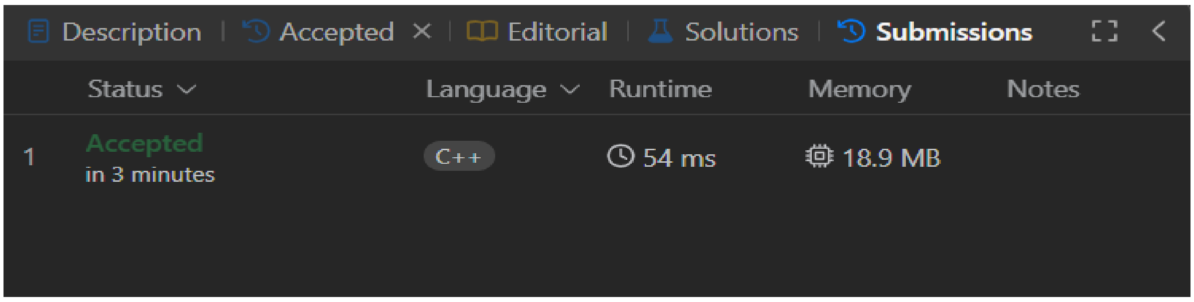


```
   Status              Language   Runtime   Memory     Notes
1  Accepted            C++        1 ms      312.2 MB
   11 minutes ago
```

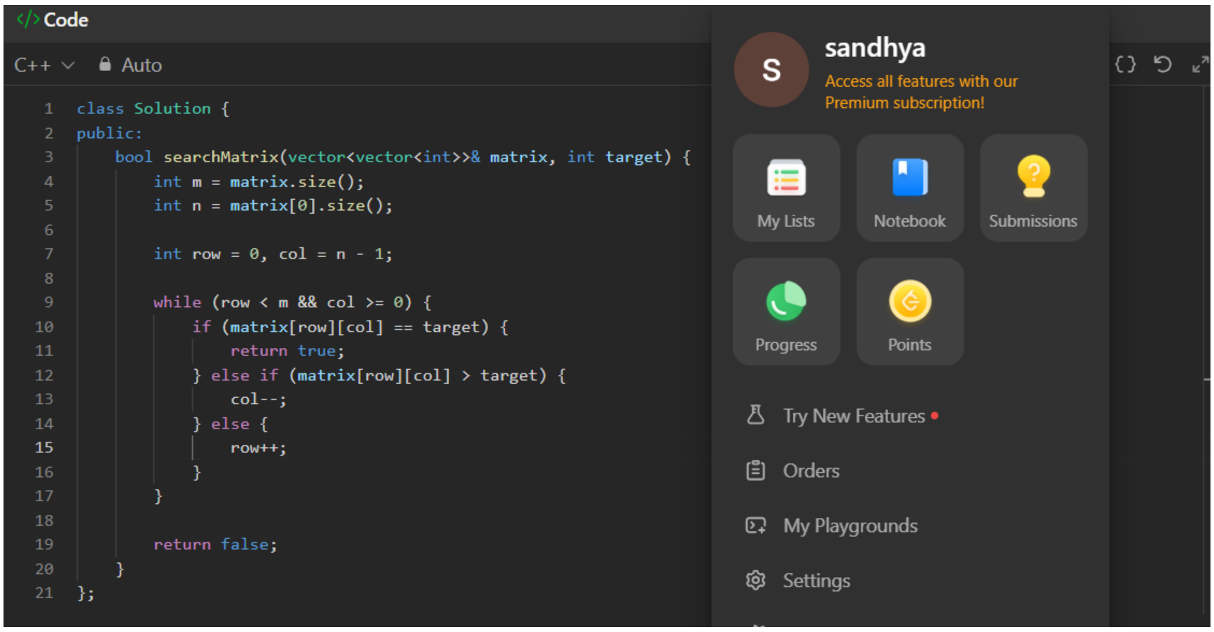


```cpp
#include <vector>
using namespace std;

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

        for (int i = 1; i < nums.size(); i++) {
            currentSum = max(nums[i], currentSum + nums[i]);
            maxSum = max(maxSum, currentSum);
        }

        return maxSum;
    }
};
```

5. Search a 2D Matrix II:

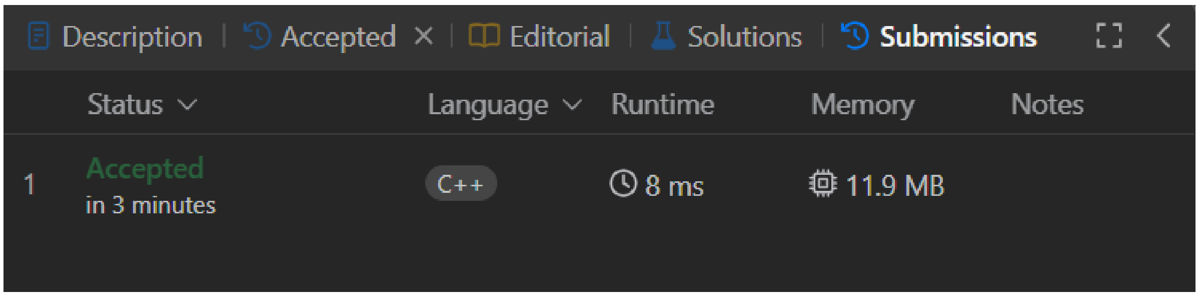


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	54 ms	18.9 MB	

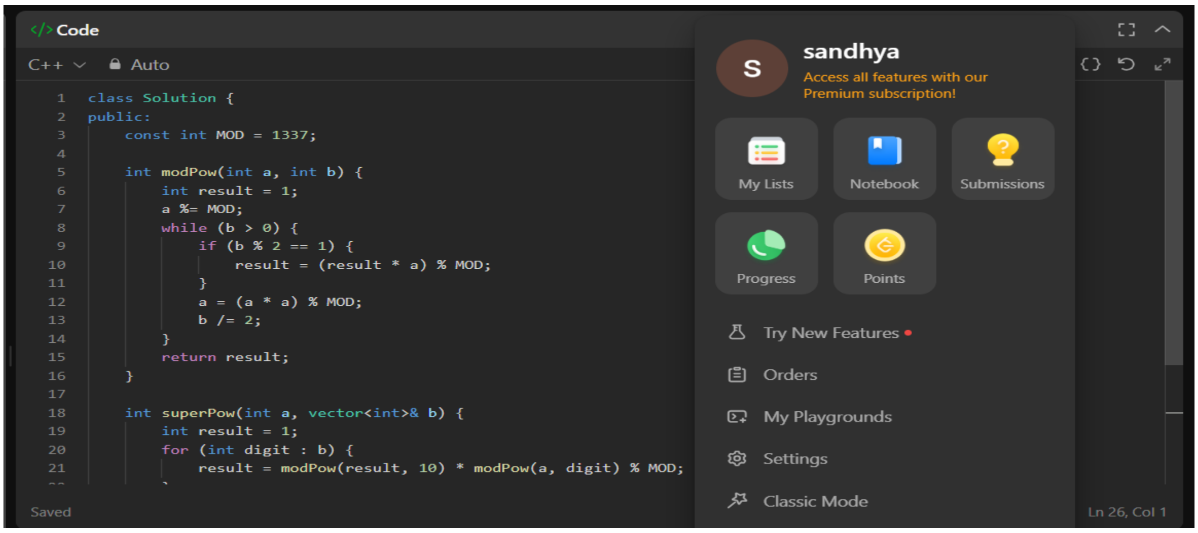


```cpp
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int m = matrix.size();
        int n = matrix[0].size();

        int row = 0, col = n - 1;

        while (row < m && col >= 0) {
            if (matrix[row][col] == target) {
                return true;
            } else if (matrix[row][col] > target) {
                col--;
            } else {
                row++;
            }
        }

        return false;
    }
};
```

6. Super Pow:

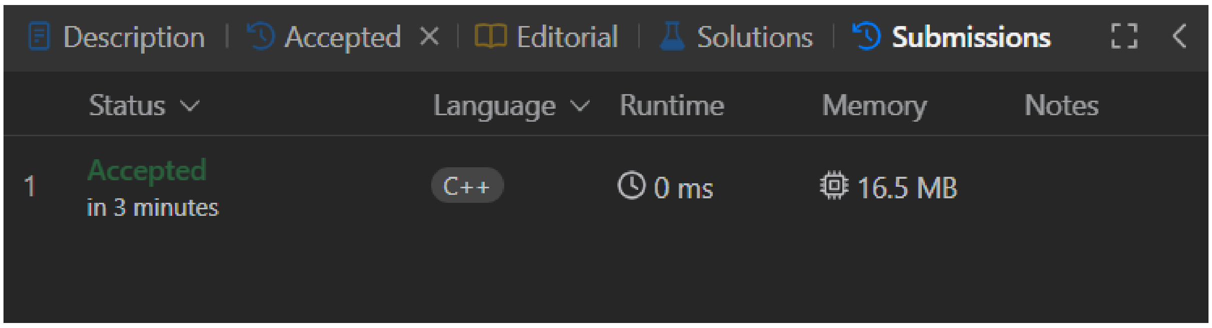


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	8 ms	11.9 MB	

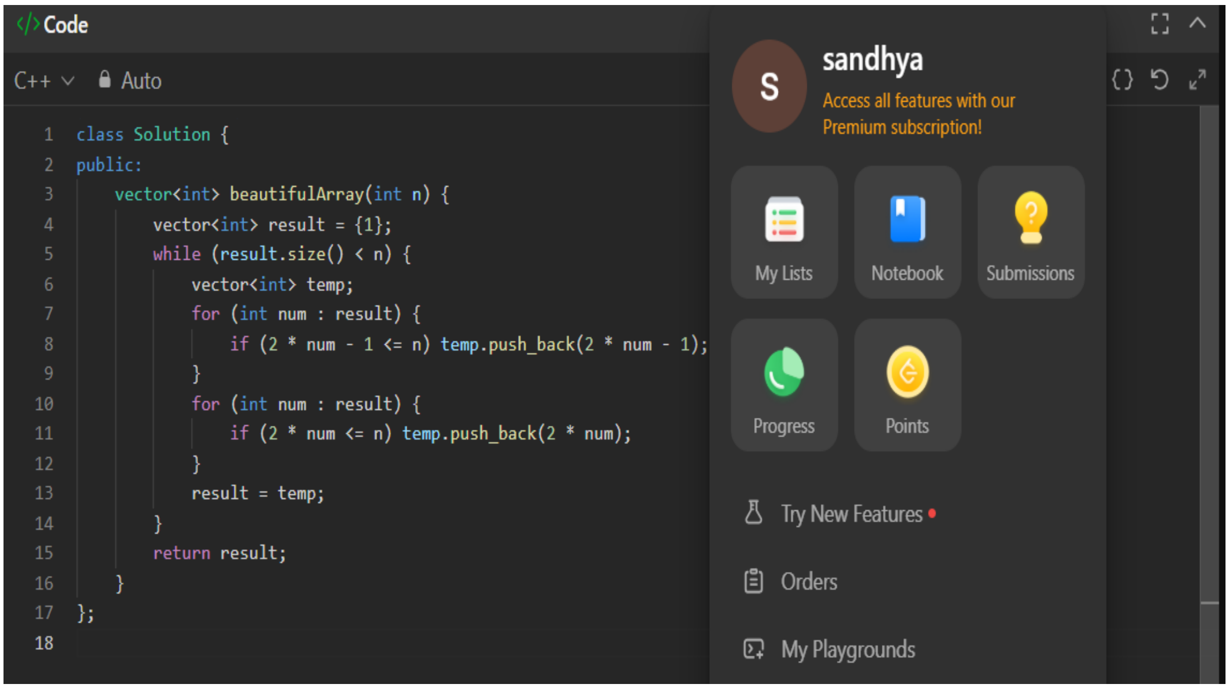


```cpp
class Solution {
public:
    const int MOD = 1337;

    int modPow(int a, int b) {
        int result = 1;
        a %= MOD;
        while (b > 0) {
            if (b % 2 == 1) {
                result = (result * a) % MOD;
            }
            a = (a * a) % MOD;
            b /= 2;
        }
        return result;
    }

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

7. Beautiful Array:

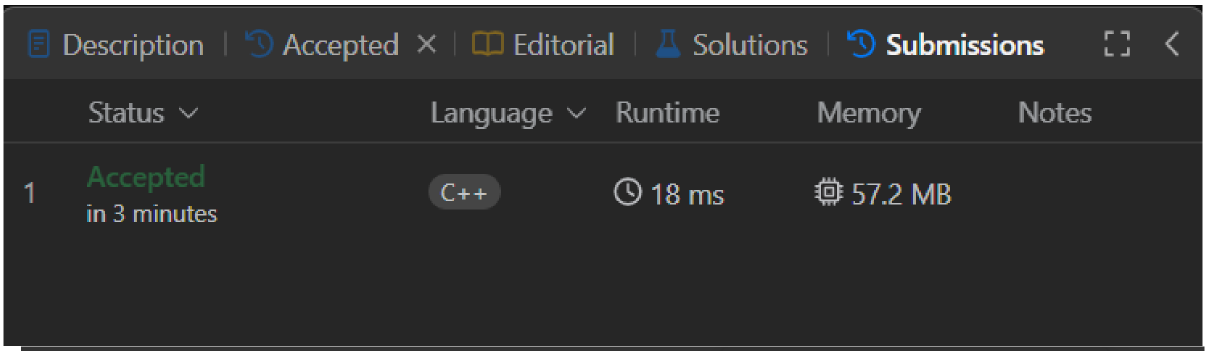


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	0 ms	16.5 MB	

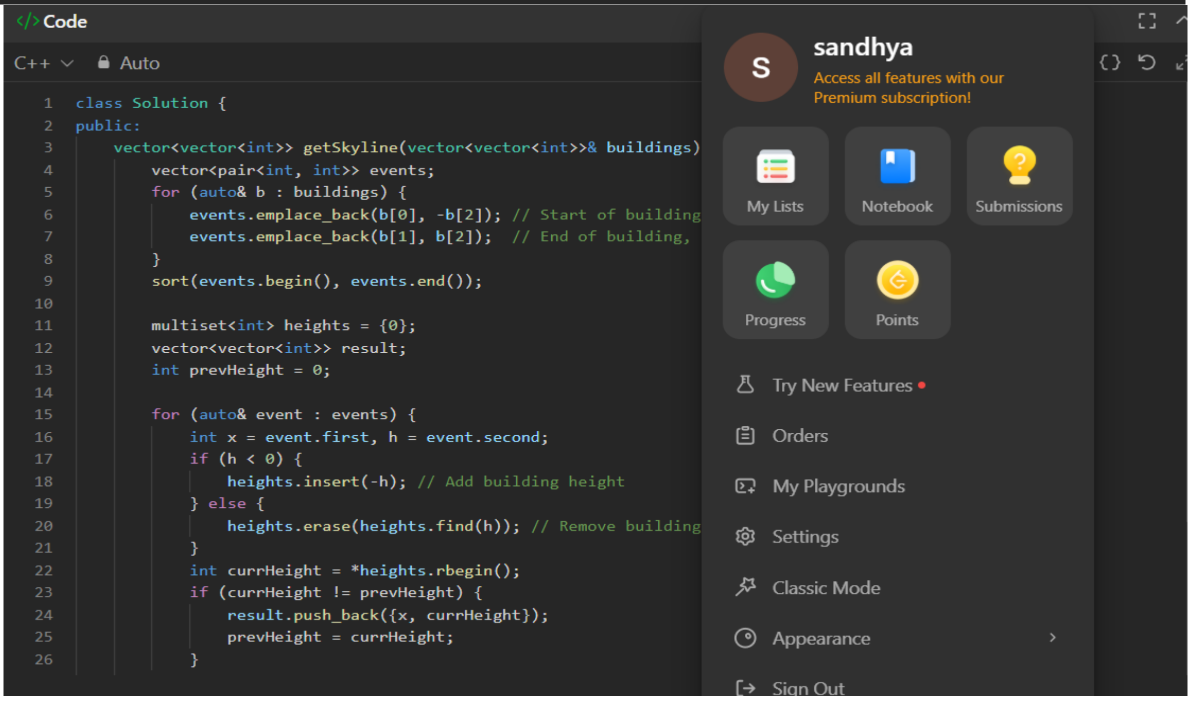


```cpp
class Solution {
public:
    vector<int> beautifulArray(int n) {
        vector<int> result = {1};
        while (result.size() < n) {
            vector<int> temp;
            for (int num : result) {
                if (2 * num - 1 <= n) temp.push_back(2 * num - 1);
            }
          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

8. The Skyline Problem:

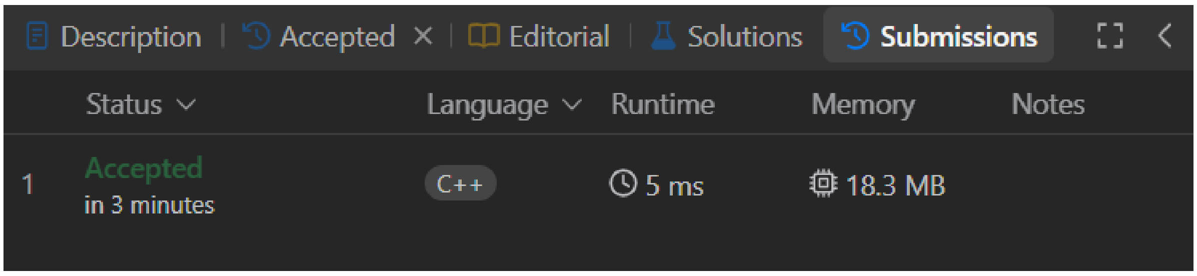


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	18 ms	57.2 MB	

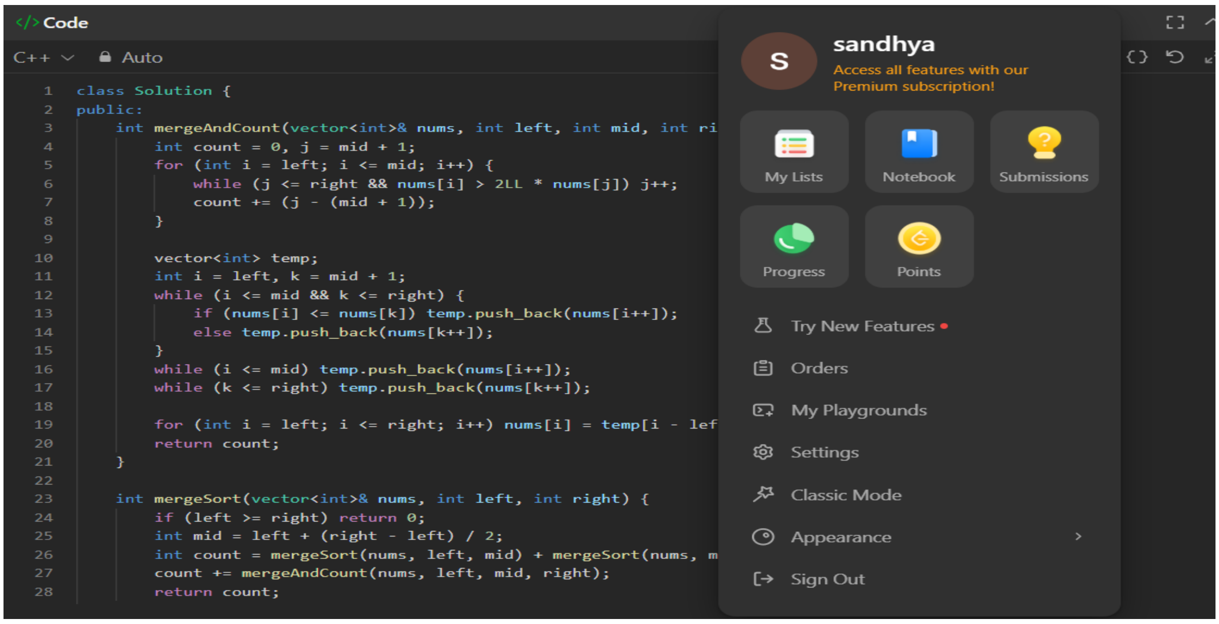


```cpp
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings)
        vector<pair<int, int>> events;
        for (auto& b : buildings) {
            events.emplace_back(b[0], -b[2]); // Start of building
            events.emplace_back(b[1], b[2]);  // End of building,
        }
        sort(events.begin(), events.end());

        multiset<int> heights = {0};
        vector<vector<int>> result;
        int prevHeight = 0;

        for (auto& event : events) {
            int x = event.first, h = event.second;
            if (h < 0) {
                heights.insert(-h); // Add building height
            } else {
                heights.erase(heights.find(h)); // Remove building
            }
            int currHeight = *heights.rbegin();
            if (currHeight != prevHeight) {
                result.push_back({x, currHeight});
                prevHeight = currHeight;
            }
```

9. Reverse Pairs:

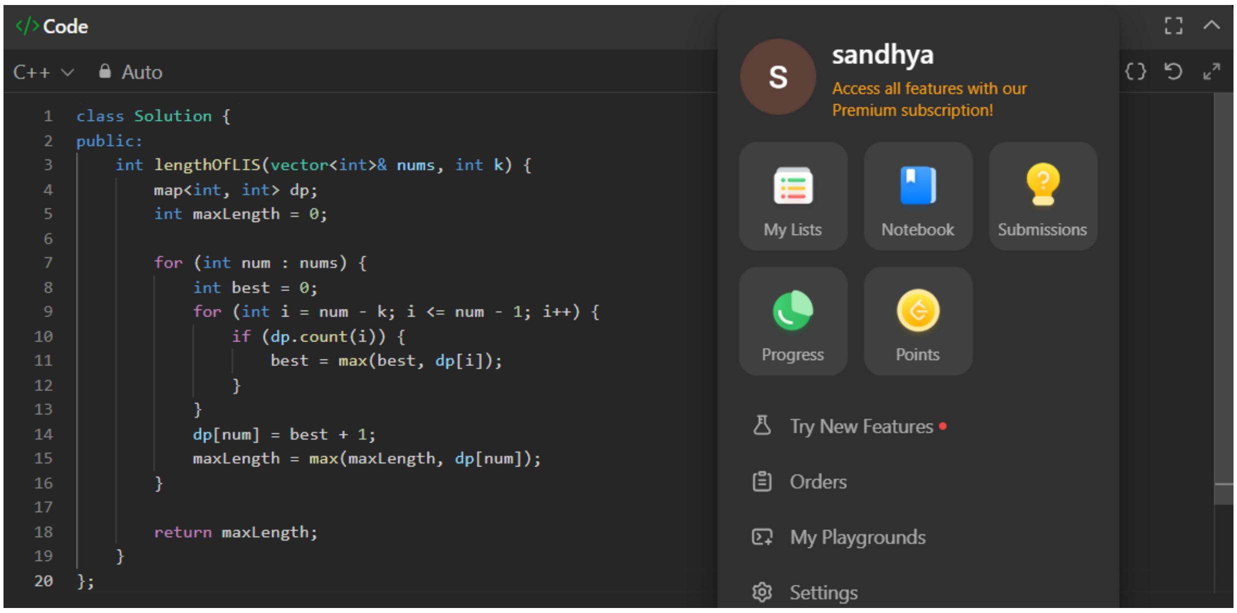


	Status	Language	Runtime	Memory	Notes
1	Accepted in 3 minutes	C++	5 ms	18.3 MB	



```cpp
class Solution {
public:
    int mergeAndCount(vector<int>& nums, int left, int mid, int ri
        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));
        }

        vector<int> temp;
        int i = left, k = mid + 1;
        while (i <= mid && k <= right) {
            if (nums[i] <= nums[k]) temp.push_back(nums[i++]);
            else temp.push_back(nums[k++]);
        }
        while (i <= mid) temp.push_back(nums[i++]);
        while (k <= right) temp.push_back(nums[k++]);

        for (int i = left; i <= right; i++) nums[i] = temp[i - lef
        return count;
    }

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

10. Longest Increasing Subsequence II:



```cpp
class Solution {
public:
    int lengthOfLIS(vector<int>& nums, int k) {
        map<int, int> dp;
        int maxLength = 0;

        for (int num : nums) {
            int best = 0;
            for (int i = num - k; i <= num - 1; i++) {
                if (dp.count(i)) {
                    best = max(best, dp[i]);
                }
            }
            dp[num] = best + 1;
            maxLength = max(maxLength, dp[num]);
        }

        return maxLength;
    }
};
```