Name-Priyanka Sharma

Uid-22BCS15114

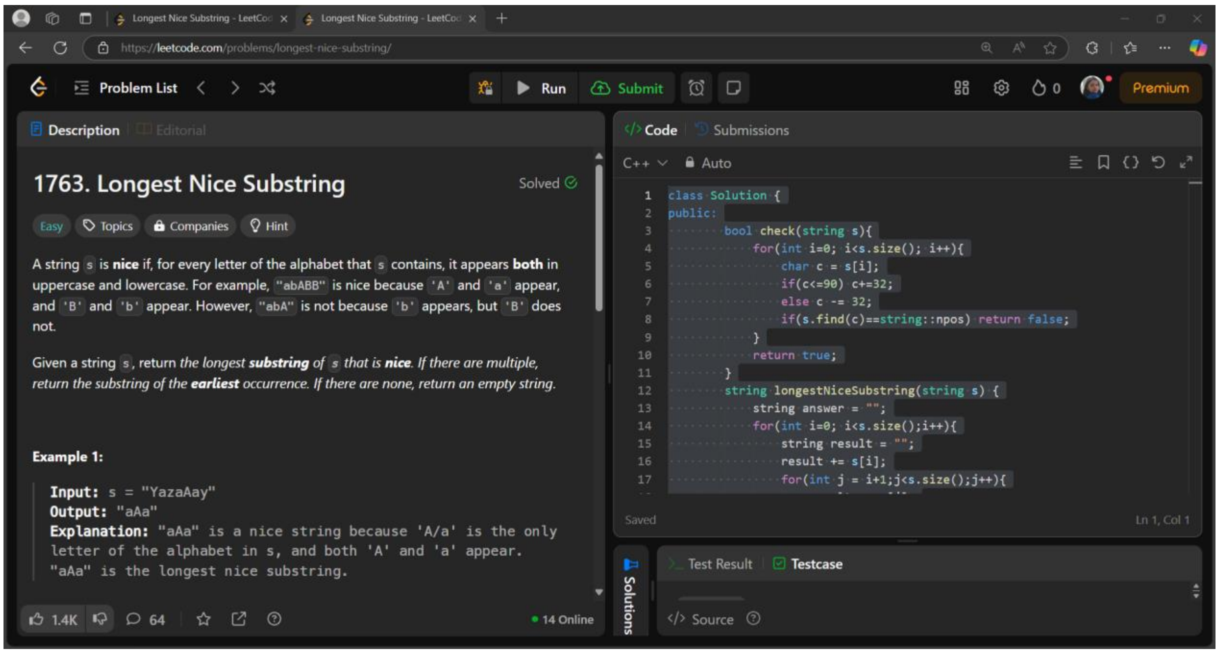
Section-608/B

AP ASSIGNMENT

1. Longest Nice substring :

```
class Solution {
public:
    bool check(string s){
      for(int i=0; i<s.size(); i++){
        char c = s[i];
        if(c<=90) c+=32;
        else c -= 32;
        if(s.find(c)==string::npos) return false;
      }
      return true;
    }
    string longestNiceSubstring(string s) {
      string answer = "";
      for(int i=0; i<s.size();i++){
        string result = "";
        result += s[i];
        for(int j = i+1;j<s.size();j++){
          result += s[j];
          if(check(result) and result.size()>answer.size()) answer = result;
        }
```

```
            }
            return answer;
        }
    };
```



https://leetcode.com/problems/longest-nice-substring/

1763. Longest Nice Substring

Solved

A string `s` is **nice** if, for every letter of the alphabet that `s` contains, it appears **both** in uppercase and lowercase. For example, `"abABB"` is nice because `'A'` and `'a'` appear, and `'B'` and `'b'` appear. However, `"abA"` is not because `'b'` appears, but `'B'` does not.

Given a string `s`, return *the longest **substring** of `s` that is **nice**. If there are multiple, return the substring of the **earliest** occurrence. If there are none, return an empty string.*

Example 1:

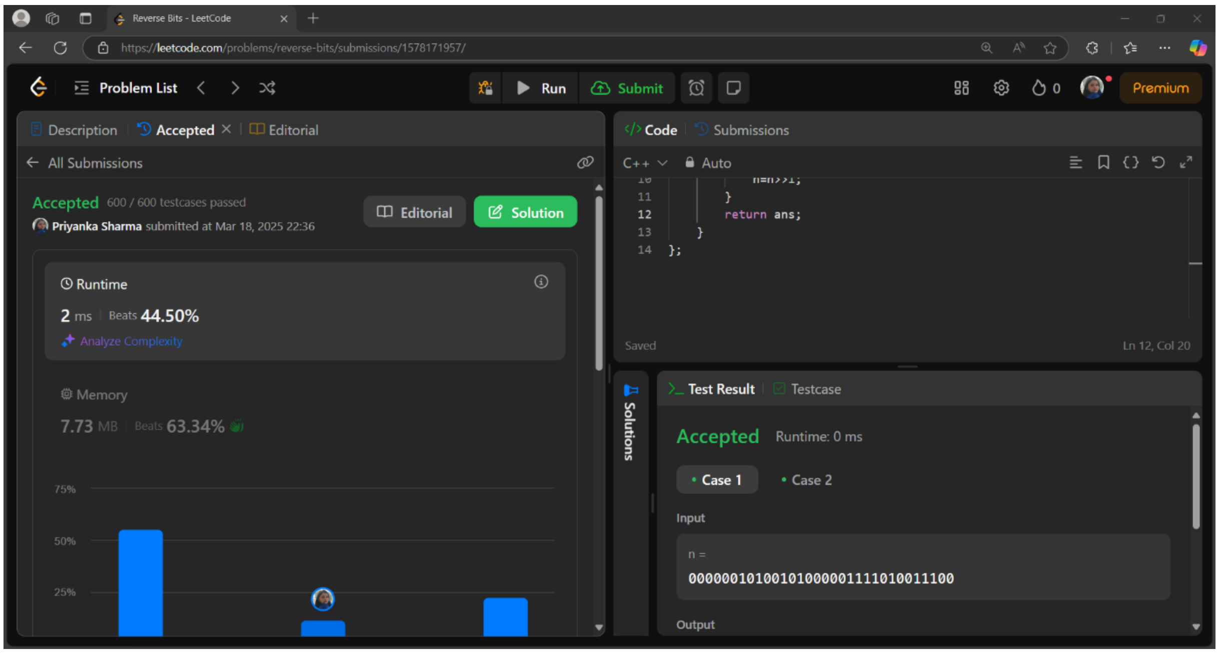
```
Input: s = "YazaAay"
Output: "aAa"
Explanation: "aAa" is a nice string because 'A/a' is the only
letter of the alphabet in s, and both 'A' and 'a' appear.
"aAa" is the longest nice substring.
```

```
class Solution {
public:
        bool check(string s){
            for(int i=0; i<s.size(); i++){
                char c = s[i];
                if(c<=90) c+=32;
                else c -= 32;
                if(s.find(c)==string::npos) return false;
            }
            return true;
        }
        string longestNiceSubstring(string s) {
            string answer = "";
            for(int i=0; i<s.size();i++){
                string result = "";
                result += s[i];
                for(int j = i+1;j<s.size();j++){
```

2.Reverse bits : class

Solution { public:

```
   uint32_t reverseBits(uint32_t n) {
uint32_t ans=0;         for (int i = 0; i <
32;  i++) {                 ans =   ans<<1;
if(n&1){              ans=ans|1;
         }
         n = n>>1;
      }
      return ans;
   }
};
```


Reverse Bits - LeetCode

https://leetcode.com/problems/reverse-bits/submissions/1578171957/

Accepted 600 / 600 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:36

Runtime

2 ms | Beats 44.50%

Memory

7.73 MB | Beats 63.34%

```
        n=n>>1;
    }
    return ans;
  }
};
```

Saved · Ln 12, Col 20

Test Result

Accepted Runtime: 0 ms

Case 1 · Case 2

Input

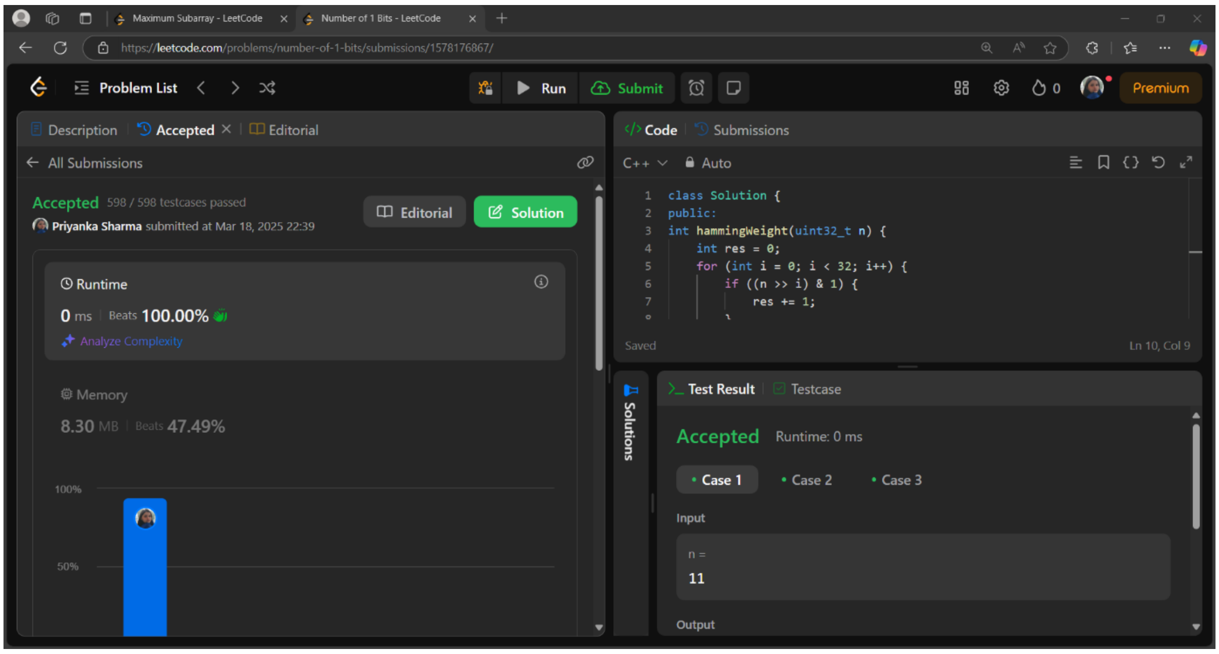
n =

```
00000010100101000001111010011100
```

Output

3.Number of 1-bits:

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


https://leetcode.com/problems/number-of-1-bits/submissions/1578176867/

Accepted 598 / 598 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:39

Runtime
0 ms | Beats 100.00%

Memory
8.30 MB | Beats 47.49%

```
class Solution {
public:
int hammingWeight(uint32_t n) {
    int res = 0;
    for (int i = 0; i < 32; i++) {
        if ((n >> i) & 1) {
            res += 1;
```

Saved   Ln 10, Col 9

Test Result

Accepted Runtime: 0 ms

Input

n =
11

Output

4.maximum of subbarray:

class Solution { public:

   int maxSubArray(vector<int>& nums) {

        int res = nums[0];

int total = 0;

        for (int n : nums) {

if (total < 0) {

total = 0;

            }

            total += n;   res

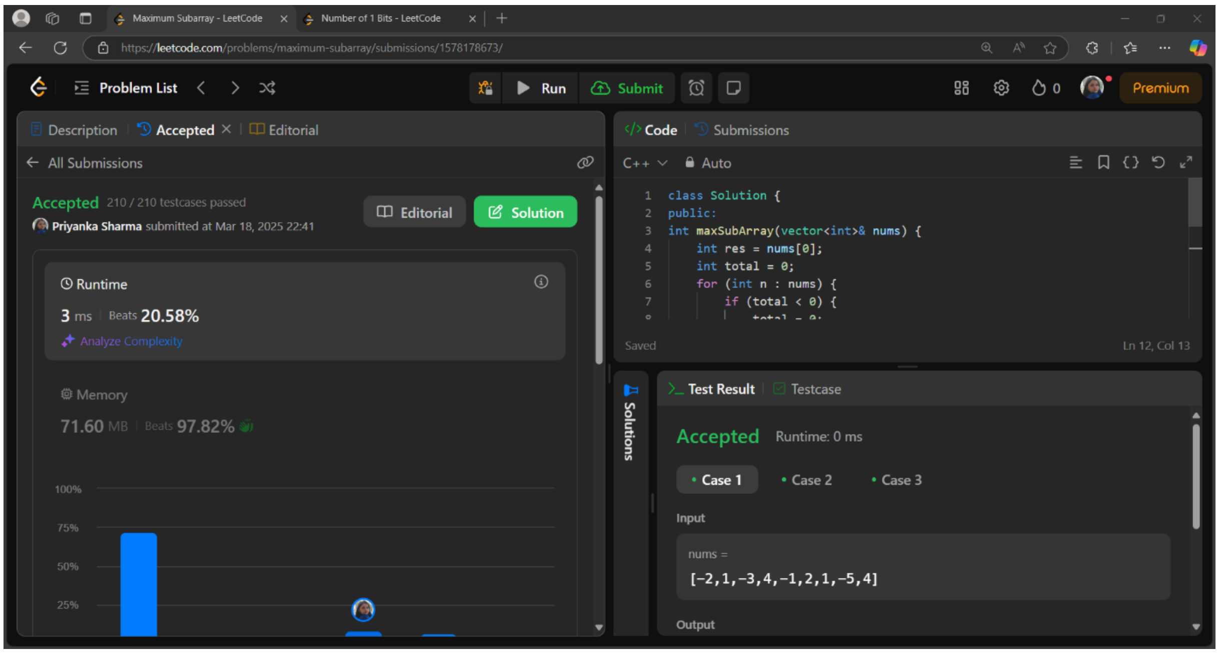
= max(res, total);

        }

        return res;

    }

};



Accepted 210 / 210 testcases passed
Priyanka Sharma submitted at Mar 18, 2025 22:41

Runtime
3 ms | Beats 20.58%
Analyze Complexity

Memory
71.60 MB | Beats 97.82%

100%
75%
50%
25%

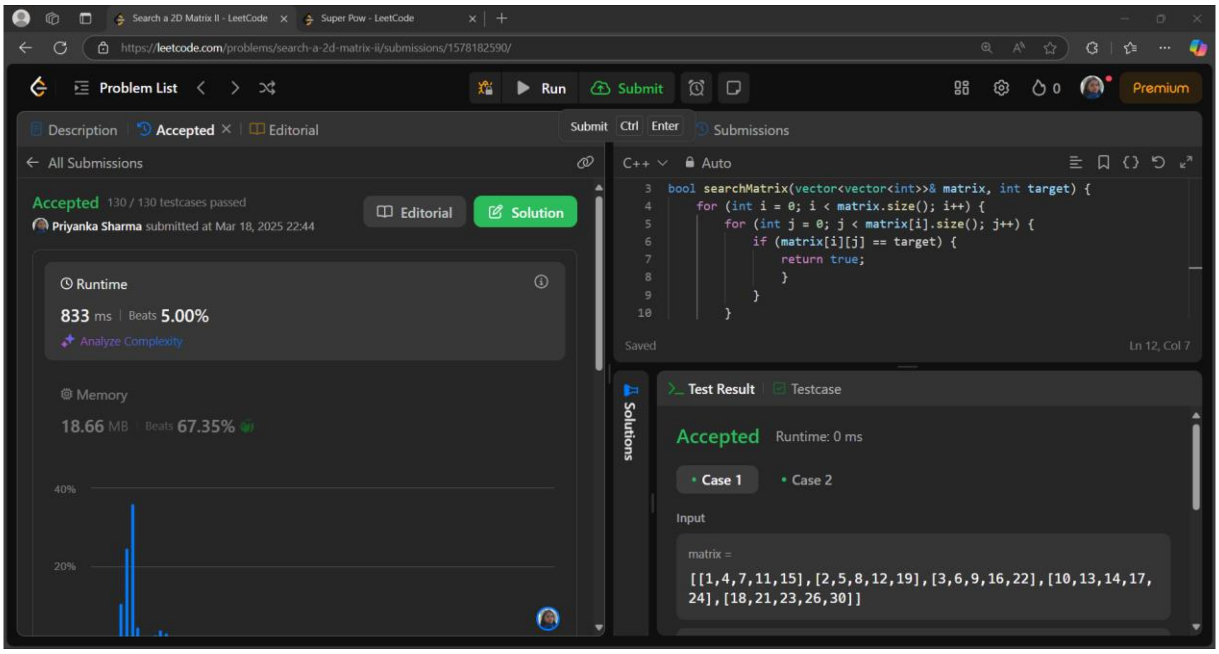
```cpp
class Solution {
public:
int maxSubArray(vector<int>& nums) {
    int res = nums[0];
    int total = 0;
    for (int n : nums) {
        if (total < 0) {
```

Saved
Ln 12, Col 13

Accepted Runtime: 0 ms
Case 1 · Case 2 · Case 3
Input
nums =
[-2,1,-3,4,-1,2,1,-5,4]
Output

5.Search a 2D matrix :

```
class Solution { public:

    bool searchMatrix(vector<vector<int>>& matrix, int target) {

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

for (int j = 0; j < matrix[i].size(); j++) {

if (matrix[i][j] == target) {                    return

true;

                }

            }

        }

        return false;

    }

};
```



Accepted 130 / 130 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:44

Runtime

833 ms | Beats 5.00%

Memory

18.66 MB | Beats 67.35%

C++ Auto

```
bool searchMatrix(vector<vector<int>>& matrix, int target) {
    for (int i = 0; i < matrix.size(); i++) {
        for (int j = 0; j < matrix[i].size(); j++) {
            if (matrix[i][j] == target) {
                return true;
            }
        }
    }
```

Accepted Runtime: 0 ms

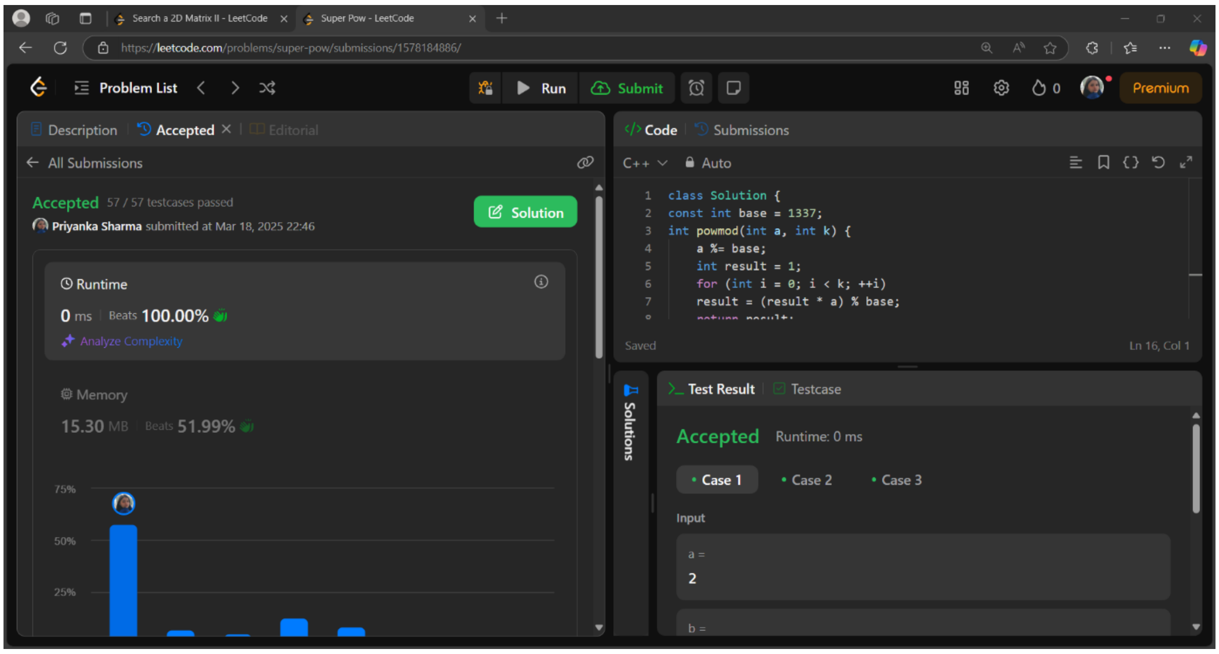
Input

matrix =

```
[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,
24],[18,21,23,26,30]]
```

6.super pow:

```
class Solution {     const int base = 1337;     int powmod(int a,
int k) //a^k mod 1337 where 0 <= k <= 10
    {
        a %= base;         int result = 1;
for (int i = 0; i < k; ++i)
result = (result * a) % base;
return result;
    }
public:
    int superPow(int a, vector<int>& b) {
if (b.empty()) return 1;         int
last_digit = b.back();
        b.pop_back();        return powmod(superPow(a, b), 10) * powmod(a,
last_digit) % base;     }

};
```


Accepted 57 / 57 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:46

Runtime

0 ms Beats 100.00%

Memory

15.30 MB Beats 51.99%

```
class Solution {
const int base = 1337;
int powmod(int a, int k) {
    a %= base;
    int result = 1;
    for (int i = 0; i < k; ++i)
    result = (result * a) % base;
```

Accepted Runtime: 0 ms

Input

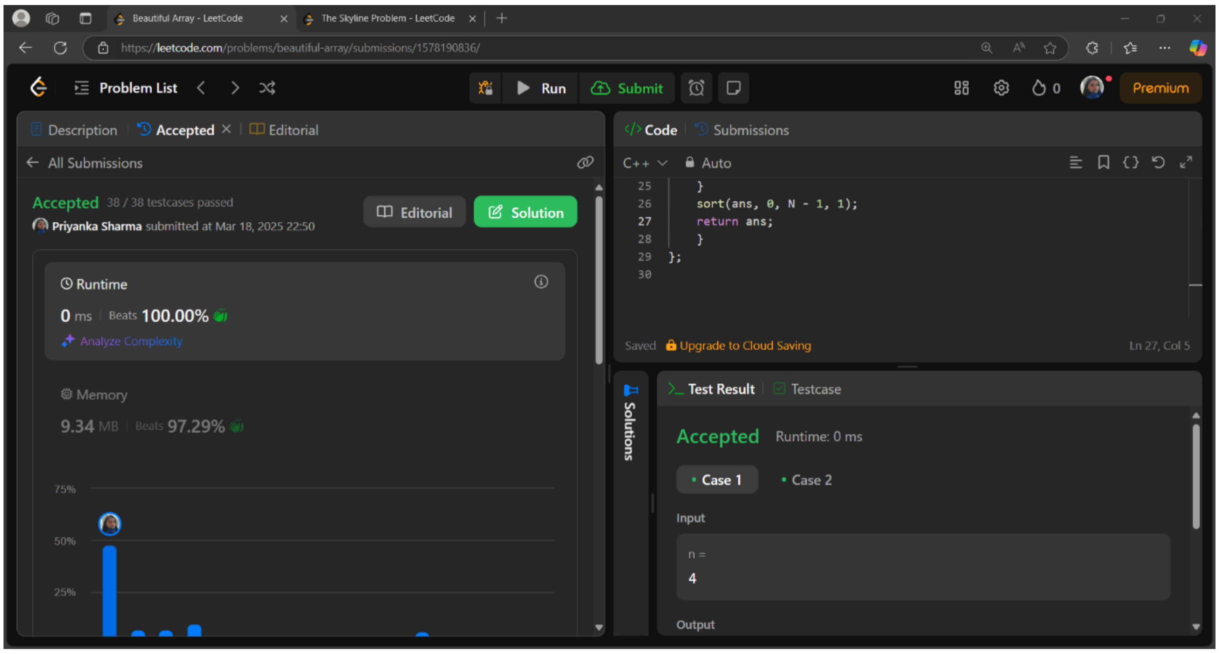
a =
2

b =

7.beautiful array :

```
class  Solution  {
public:
  int partition(vector<int> &v, int start, int end, int mask)
  {      int j = start;      for(int i =
start; i <= end; i++)
    {
      if((v[i] & mask) != 0)
      {
swap(v[i], v[j]);
j++;
      }
    }
    return j;
  }

  void sort(vector<int> & v, int start, int end, int mask)
```

```
   {
       if(start >= end) return;        int mid =
partition(v, start, end, mask);        sort(v,
start, mid - 1, mask << 1);        sort(v, mid,
end, mask << 1);
   }

   vector<int> beautifulArray(int N) {
vector<int> ans;        for(int i = 0; i < N; i++)
ans.push_back(i + 1);        sort(ans, 0, N - 1, 1);
return ans;
   }
};
```



Beautiful Array - LeetCode | The Skyline Problem - LeetCode

https://leetcode.com/problems/beautiful-array/submissions/1578190836/

Problem List | Run | Submit | Premium

Description | Accepted | Editorial

All Submissions

Accepted 38 / 38 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:50

Editorial | Solution

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

9.34 MB | Beats 97.29%

Code | Submissions

C++ | Auto

```
        }
        sort(ans, 0, N - 1, 1);
        return ans;
    }
};
```

Saved | Upgrade to Cloud Saving | Ln 27, Col 5

Test Result | Testcase

Accepted Runtime: 0 ms

Case 1 | Case 2

Input

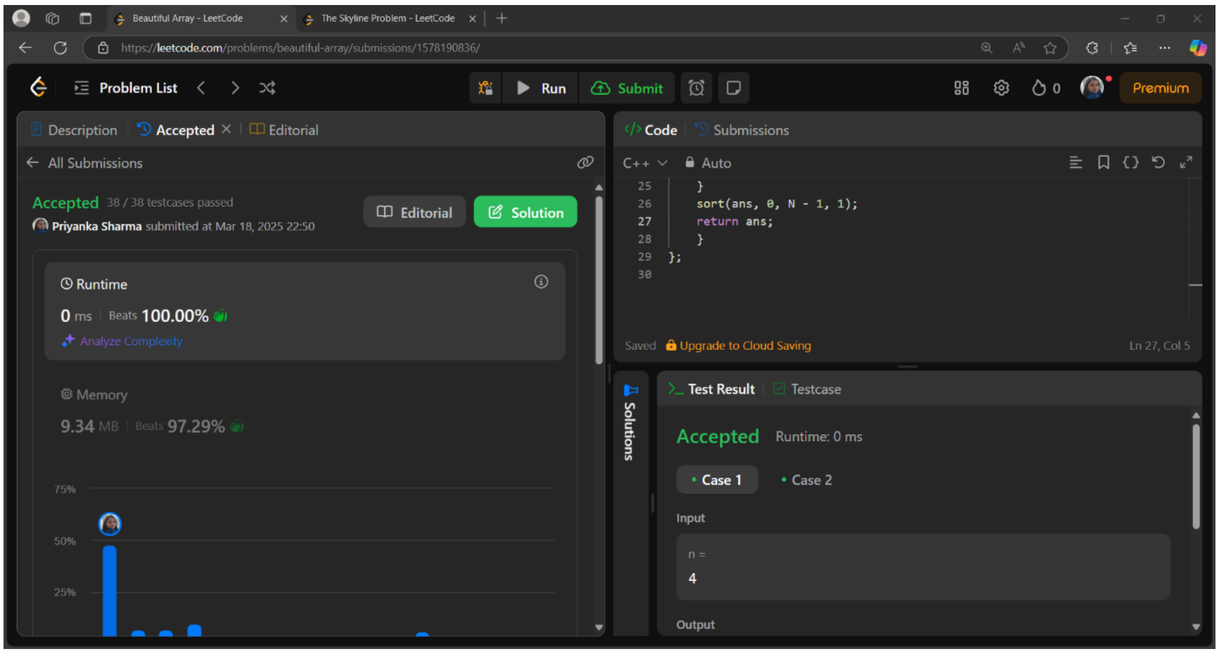
n =
4

Output

8.the skyline problem:

```
class Solution { public:
   vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
vector<vector<int>> ans;        multiset<int> pq{0};
```

```
        vector<pair<int, int>> points;

        for(auto b: buildings){
points.push_back({b[0], -b[2]});
points.push_back({b[1], b[2]});
        }

        sort(points.begin(), points.end());

        int ongoingHeight = 0;

        // points.first = x coordinate, points.second = height
for(int i = 0; i < points.size(); i++){            int currentPoint
= points[i].first;            int heightAtCurrentPoint =
points[i].second;

            if(heightAtCurrentPoint < 0){                pq.insert(-
heightAtCurrentPoint);
            } else {
                pq.erase(pq.find(heightAtCurrentPoint));
            }

            // after inserting/removing heightAtI, if there's a change
auto pqTop = *pq.rbegin();            if(ongoingHeight != pqTop){
ongoingHeight = pqTop;                ans.push_back({currentPoint,
ongoingHeight});
            }
        }


        return ans;
    }
};
```



https://leetcode.com/problems/beautiful-array/submissions/1578190836/

Accepted 38 / 38 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 22:50

Runtime

0 ms | Beats 100.00%

Memory

9.34 MB | Beats 97.29%

```
    }
    sort(ans, 0, N - 1, 1);
    return ans;
    }
};
```

Accepted Runtime: 0 ms

Case 1 Case 2

Input

n =
4

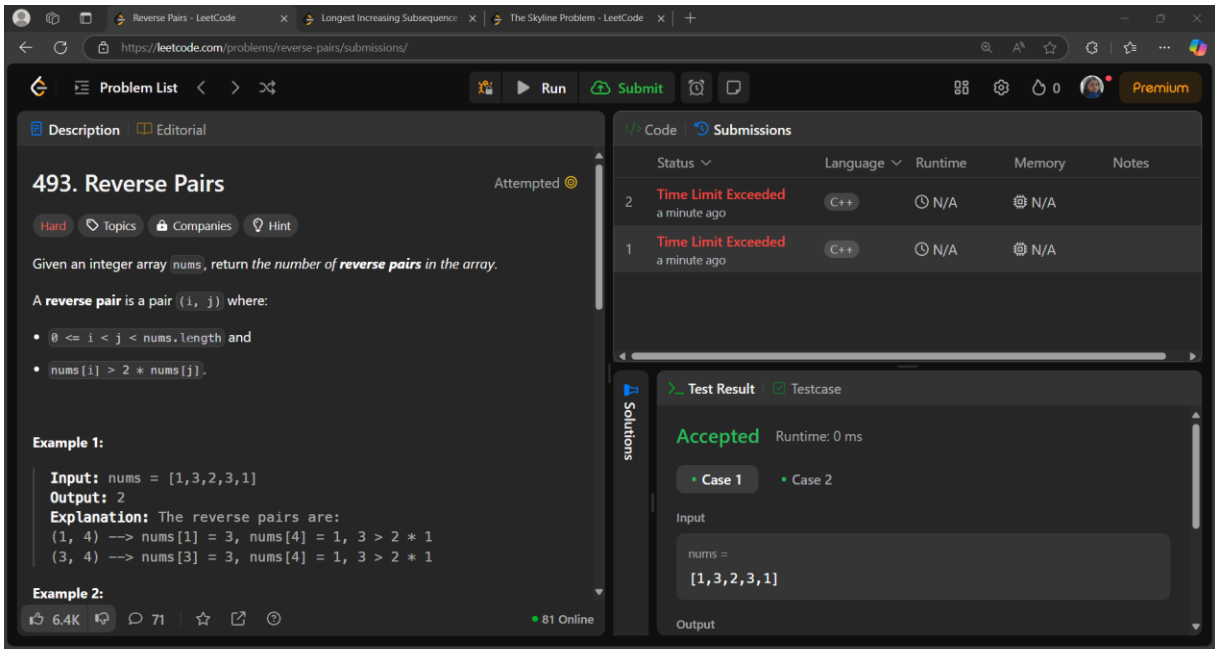
Output

9.Reverse pairs : class

Solution

```
{
    int get_pairs(vector<int>& vct , long long int x)
    {
        //sort(vct.begin() , vct.end());
int size = vct.size();        int low =
0;        int high = size - 1;        int
ans = -1;        while(low <= high)
        {
            int mid = high - (high - low) / 2;
int ele = vct[mid];            if(ele > x)
            {
                ans = mid;
high = mid - 1;
            }
else
            {
                low = mid  + 1;
            }
        }
        if(ans == -1) return 0;
return vct.size() - ans;
    }

    // void print_vector(vector<int>& nums)
    // {
    //     cout<<endl;
    //     for(auto it : nums)
    //     {
    //         cout<<" "<<it;
    //     }
    //     cout<<endl;
    // }

public:     int reversePairs(vector<int>&
nums)
    {
        vector<int> vct;
int counter = 0;
for(auto it : nums)
        {
            long long int x = 1LL * 2 * it;
counter += get_pairs(vct , x);
int low = 0;            int high =
vct.size();            int ans = vct.size();
while(low < high)
            {
                int mid = low + (high - low) / 2;
if(vct[mid] >= it)
                {
ans = mid;
high = mid;
                }
                else
                {
                    low = mid + 1;
                }
            }
            vct.insert(vct.begin() + ans , it);
            //print_vector(vct);
        }

        return counter;
    }
```

};


Reverse Pairs - LeetCode | Longest Increasing Subsequence | The Skyline Problem - LeetCode

https://leetcode.com/problems/reverse-pairs/submissions/

Problem List | Run | Submit | Premium

Description | Editorial

493. Reverse Pairs Attempted

Hard | Topics | Companies | Hint

Given an integer array `nums`, return *the number of* ***reverse pairs*** *in the array.*

A **reverse pair** is a pair `(i, j)` where:

- `0 <= i < j < nums.length` and
- `nums[i] > 2 * nums[j]`.

Example 1:

```
Input: nums = [1,3,2,3,1]
Output: 2
Explanation: The reverse pairs are:
(1, 4) --> nums[1] = 3, nums[4] = 1, 3 > 2 * 1
(3, 4) --> nums[3] = 3, nums[4] = 1, 3 > 2 * 1
```

Example 2:

6.4K | 71 | 81 Online

Code | Submissions

	Status	Language	Runtime	Memory	Notes
2	Time Limit Exceeded a minute ago	C++	N/A	N/A	
1	Time Limit Exceeded a minute ago	C++	N/A	N/A	

Test Result | Testcase

Accepted Runtime: 0 ms

Case 1 | Case 2

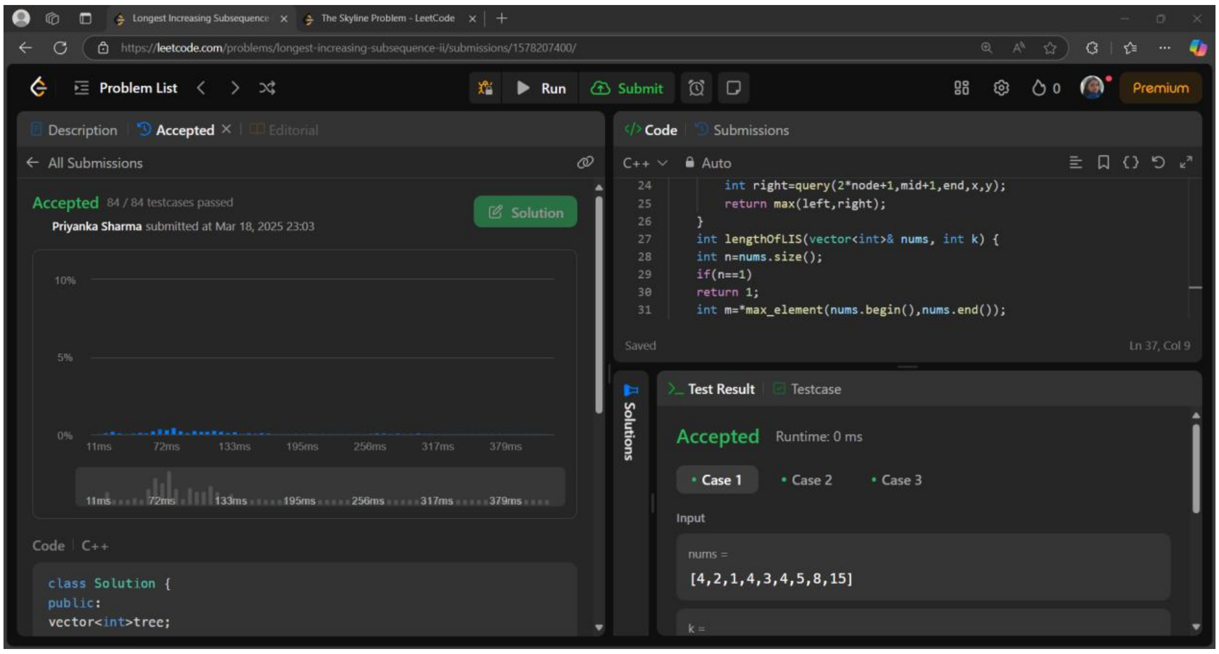
Input

nums =
[1,3,2,3,1]

Output

10.longest increasing substring :

```
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];
   }
};
```



Accepted 84 / 84 testcases passed

Priyanka Sharma submitted at Mar 18, 2025 23:03

Code | C++

```
class Solution {
public:
vector<int>tree;
```

```
        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());
```

Accepted Runtime: 0 ms

Case 1 • Case 2 • Case 3

Input

nums =

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

k =