

# Advanced Programming

## Assignment 4

Name-Rahul Gupta

UID-22BCS10469

### Q1.) Longest Nice Substring

```
class Solution {
private:
    bool isNice(string& str){
        for(char c:str){
            if(islower(c)&&str.find(toupper(c))==string::npos){
                return false;
            }
            if(isupper(c)&&str.find(tolower(c))==string::npos){
                return false;
            }
        }
        return true;
    }
public:
    string longestNiceSubstring(string s) {
        string ans="";
        int n=s.length();
        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)){
                    if(sub.length()>ans.length()){
```

```

        ans=sub;
    }
}
}
}
return ans;
}
};

```

Problem List

Run
Submit

54
Premium

Description
Editorial
Solutions
Submissions
Accepted

All Submissions

Accepted 73 / 73 testcases passed  
Rahul Gupta submitted at Feb 25, 2025 12:03

Solution

Runtime

23 ms | Beats 19.65%

Analyze Complexity

Memory

13.86 MB | Beats 75.35%

Code | C++

Code

C++
Auto

```

1 class Solution {
2 private:
3     bool isNice(string& str){
4         for(char c:str){
5             if(islower(c)&&str.find(toupper(c))==string::npos){
6                 return false;
7             }
8             if(isupper(c)&&str.find(tolower(c))==string::npos){
9                 return false;
10            }
11        }
12        return true;
13    }
14 public:
15     string longestNiceSubstring(string s) {
16         string ans="";
17         int maxLen=0;

```

Saved
Ln 15, Col 44

Testcase
Test Result

Case 1 Case 2 Case 3 +

s =

"YazaAay"

Source

## Q2.) Reverse Bits

```
class Solution {
```

```
public:
```

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

```
        uint32_t result = 0;
```

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

```
            int bit = n & 1;
```

```
            result = (result << 1) | bit;
```

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

```
        }
```

```
        return result;
```

```
    }
```

```
};
```

The screenshot displays a coding platform interface with the following components:

- Problem List:** Shows the current problem and navigation options.
- Code Editor:** Contains the C++ solution for the 'Reverse Bits' problem. The code is as follows:

```
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             int bit = n & 1;
7             result = (result << 1) | bit;
8             n = n >> 1;
9         }
10        return result;
11    }
12};
```
- Runtime/Memory Analysis:** Shows the solution is 'Accepted' with 600/600 testcases passed. The runtime is 0 ms (Beats 100.00%) and memory is 7.69 MB (Beats 87.34%). A bar chart shows the performance relative to other submissions.
- Testcase Section:** Displays a test case with input `n = 00000010100101000001111010011100`.

### Q3.) 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;  
    }  
};
```

The screenshot displays a C++ IDE interface for a problem titled "Number of 1 Bits". The left sidebar shows the "Accepted" status with 598 / 598 testcases passed, submitted by "Rahul Gupta" on Mar 18, 2025 10:05. The "Runtime" section indicates 0 ms and 100.00% beats, while the "Memory" section shows 8.08 MB and 98.45% beats. A bar chart visualizes the performance across different test cases. The main editor shows the C++ code for the `hammingWeight` function. The bottom right panel displays the "Testcase" tab with "Case 1" selected, showing the input `n = 11`.

**Runtime:** 0 ms | Beats 100.00%

**Memory:** 8.08 MB | Beats 98.45%

```
1 class Solution {  
2 public:  
3     int hammingWeight(uint32_t n) {  
4         int res = 0;  
5         for (int i = 0; i < 32; i++) {  
6             if ((n >> i) & 1) {  
7                 res += 1;  
8             }  
9         }  
10        return res;  
11    }  
12 };
```

**Testcase:** Case 1 | Case 2 | Case 3 | +

n = 11

#### Q4.) Maximum Subarray

```
class Solution {
```

```
public:
```

```
    int maxSubArray(vector<int>& nums) {
```

```
        int sum = 0, maxSum = INT_MIN;
```

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

```
            sum += nums[i];
```

```
            if (sum > maxSum) maxSum = sum;
```

```
            if (sum < 0) sum = 0;
```

```
        }
```

```
        return maxSum;
```

```
    }
```

```
};
```

**Accepted** 210 / 210 testcases passed  
Rahul Gupta submitted at Mar 04, 2025 11:45

**Runtime**  
0 ms | Beats 100.00%  
[Analyze Complexity](#)

**Memory**  
71.81 MB | Beats 18.41%

Bar chart showing performance comparison (0% to 100%):

Performance	Count
0%	1
1ms	1
2ms	1
3ms	1
4ms	1
5ms	1

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int sum = 0, maxSum = INT_MIN;
        for (int i = 0; i < nums.size(); i++) {
            sum += nums[i];
            if (sum > maxSum) maxSum = sum;
            if (sum < 0) sum = 0;
        }
        return maxSum;
    }
};
```

**Testcase** | Test Result

Case 1 | Case 2 | Case 3 | +

nums =

[-2, 1, -3, 4, -1, 2, 1, -5, 4]

Code | C++

```
class Solution {
```

## Q5.) [Search a 2D Matrix II](#)

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

The screenshot displays a coding platform interface with the following components:

- Top Bar:** Includes navigation icons, a 'Problem List' tab, and buttons for 'Run', 'Submit', and 'Premium'.
- Left Panel:**
  - Accepted:** 130 / 130 testcases passed.
  - Editorial:** A green button labeled 'Solution'.
  - Runtime:** 63 ms | Beats 24.28%.
  - Memory:** 18.66 MB | Beats 67.35%.
  - Graph:** A bar chart showing performance metrics across different time intervals (4ms, 173ms, 343ms, 513ms).
- Code Editor:** Displays the C++ code for the solution, with line numbers 1 through 15. The code is saved.
- Testcase Section:**
  - Case 1:** 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]]. target = 5.
  - Test Result:** Shows the execution result for the test case.

## Q6.) [Super Pow](#)

```
class Solution {
```

```
private:
```

```
    int solve(int base, int power, int mod) {
```

```
        int ans = 1;
```

```
        while (power > 0) {
```

```
            if (power & 1) {
```

```
                ans = (ans * base) % mod; }
```

```
            base = (base * base) % mod;
```

```
            power >>= 1; }
```

```
        return ans; }
```

```
public:
```

```
    int superPow(int a, vector<int>& b) {
```

```
        a%=1337;
```

```
        int n = b.size();
```

```
        int m = 1140;
```

```
        int expi = 0;
```

```
        for(int i : b){
```

```
            expi = (expi*10+i)%m; }
```

```
        if (expi == 0) {
```

```
            expi = m; }
```

```
        return solve(a,expi,1337); }}
```

The screenshot shows a C++ IDE interface for the 'Super Pow' problem. The left pane displays the problem description, accepted status, and runtime/memory statistics. The right pane shows the C++ code and test cases.

**Problem List:** Accepted 57 / 57 testcases passed. Rahul Gupta submitted at Mar 18, 2025 11:34.

**Runtime:** 0 ms | Beats 100.00%.

**Memory:** 15.34 MB | Beats 15.09%.

**Code:**

```
14 public:
15     int superPow(int a, vector<int>& b) {
16         a%=1337;
17         int n = b.size();
18         int m = 1140;
19         int expi = 0;
20         for(int i : b){
21             expi = (expi*10+i)%m;
22         }
23         if (expi == 0) {
24             expi = m;
25         }
26         return solve(a,expi,1337);
27     }
28 }
```

**Testcase:** Case 1, Case 2, Case 3.

**Test Result:**

a = 2

b = [3]

## Q7.) Beautiful Array

```
class Solution {
```

```
public:
```

```
vector<int> beautifulArray(int n) {
```

```
    vector<int>ans;
```

```
    ans.push_back(1);
```

```
    while(ans.size()<n){
```

```
        vector<int>temp;
```

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

```
            if(ans[i]*2-1<=n){
```

```
                temp.push_back(ans[i]*2-1); }}
```

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

```
            if(ans[i]*2<=n){
```

```
                temp.push_back(ans[i]*2); }}
```

```
        ans=temp; }
```

```
    return ans; } };
```

The screenshot displays a coding platform interface for a C++ solution to the 'Beautiful Array' problem. The solution is accepted, with a runtime of 0 ms and memory of 10.06 MB. The code is displayed in the 'Code' tab, and the 'Testcase' tab shows a test case with n=4.

**Runtime:** 0 ms | Beats 100.00%

**Memory:** 10.06 MB | Beats 58.40%

**Code:**

```
1 class Solution {
2 public:
3     vector<int> beautifulArray(int n) {
4         vector<int>ans;
5         ans.push_back(1);
6         while(ans.size()<n){
7             vector<int>temp;
8             for(int i=0;i<ans.size();i++){
9                 if(ans[i]*2-1<=n){
10                     temp.push_back(ans[i]*2-1);
11                 }
12             }
13             for(int i=0;i<ans.size();i++){
14                 if(ans[i]*2<=n){
15                     temp.push_back(ans[i]*2);
16                 }
17             }
18             ans=temp;
19         }
20         return ans;
21     }
22 }
```

**Testcase:** Case 1

n = 4



## Q8.) The Skyline Problem

```
class Solution {  
  
public:  
  
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {  
  
        map<int,vector<int> >m;  
  
        for (auto x:buildings){  
  
            m[x[0]].push_back(x[2]);  
  
            m[x[1]].push_back(x[2]*-1); }  
  
        multiset<int> s;  
  
        vector<vector<int>> ans;  
  
        int prev=-1;  
  
        for(auto x:m){  
  
            for(auto y:x.second){  
  
                if(y>0) s.insert(y);  
  
                else if(y<0) s.erase(s.find(y*-1)); }  
  
            if(s.size() && *s.rbegin()!=prev) ans.push_back({x.first,*s.rbegin()}),prev=*s.rbegin();  
  
            if(!s.size()) ans.push_back({x.first,0}),prev=0; }  
  
        return ans; } };
```

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

- Problem List:** Shows the problem name and status (Accepted).
- Code Editor:** Contains the C++ code for the Skyline Problem solution.
- Runtime:** 38 ms, Beats 25.92%.
- Memory:** 35.18 MB, Beats 20.38%.
- Testcase:** Case 1, buildings = [[2,9,10],[3,7,15],[5,12,12],[15,20,10],[19,24,8]].

```
class Solution {  
public:  
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {  
        map<int,vector<int> >m;  
        for (auto x:buildings){  
            m[x[0]].push_back(x[2]);  
            m[x[1]].push_back(x[2]*-1); }  
        multiset<int> s;  
        vector<vector<int>> ans;  
        int prev=-1;  
        for(auto x:m){  
            for(auto y:x.second){  
                if(y>0) s.insert(y);  
                else if(y<0) s.erase(s.find(y*-1)); }  
            if(s.size() && *s.rbegin()!=prev) ans.push_back({x.first,*s.rbegin()}),prev=*s.rbegin();  
            if(!s.size()) ans.push_back({x.first,0}),prev=0; }  
        return ans; } };
```

### Q9.) [Reverse Pairs](#)

```
class Solution {  
    int get_pairs(vector<int>& vct , long long int x) {  
        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; }  
}
```

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); }

return counter; }

};

```

The screenshot displays the LeetCode submission page for the 'Reverse Pairs' problem. The submission is marked as 'Accepted' with 140/140 testcases passed. The user 'Rahul Gupta' submitted it on Mar 18, 2025 at 12:06. The performance metrics show a runtime of 774 ms (beating 5.02%) and memory usage of 53.09 MB (beating 94.27%).

The C++ code implements a binary search approach to count reverse pairs. It defines a function `reversePairs` that takes a vector of integers and returns the number of reverse pairs. The code uses a recursive function `count` to find the number of elements greater than the current element in the sorted array.

```

14
15         ans = mid;
16         high = mid - 1;
17     }
18     else
19     {
20         low = mid + 1;
21     }
22 }
23 if(ans == -1) return 0;
24 return vct.size() - ans;
25 }
26
27 public:
28     int reversePairs(vector<int>& nums)
29     {
30         vector<int> vct;

```

The test case shows the input array `nums = [1,3,2,3,1]`.

<https://leetcode.com>

### Q10.) 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];

}

};

```

Accepted 84 / 84 testcases passed  
 Rahul Gupta submitted at Mar 18, 2025 12:09

**Runtime**  
 62 ms | Beats 86.63%

**Memory**  
 59.85 MB | Beats 77.37%

**Code**

```

C++
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];
}

```

**Testcase**

Case 1 Case 2 Case 3 +

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

k =  
 3