

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Assignment-4

**Student Name:** Ayush Dixit

**Branch:** CSE

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

**Subject Name:** AP Lab

**UID:** 22BCS11401

**Section/Group:** IOT-609-B

**Date:** 17-03-2025

**Subject Code:** 22CSP-351

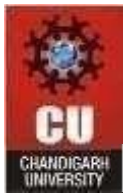
### **Ques 1: Longest Nice Substring**

**Code: -**

```
class Solution {  
public:  
    string longestNiceSubstring(string s) {  
        if (s.size() < 2) return "";  
        unordered_set<char> st(s.begin(), s.end());  
        for (int i = 0; i < s.size(); i++) {  
            if (st.find(tolower(s[i])) != st.end() && st.find(toupper(s[i])) != st.end()) continue;  
            string left = longestNiceSubstring(s.substr(0, i));  
            string right = longestNiceSubstring(s.substr(i + 1));  
            if (left.size() >= right.size()) return left;  
            else return right; }  
        return s; } };
```

**Submission: -**

The screenshot displays a code submission interface. On the left, the 'Accepted' status is shown with 73/73 testcases passed. The submission was made by 'AnanyaKamwar' on Mar 17, 2025, at 13:29. The runtime is 7 ms, beating 64.65% of solutions, and the memory usage is 14.14 MB, beating 63.14%. A bar chart shows the performance comparison. On the right, the C++ code is displayed, which is the same as the one provided in the assignment. The test case 'YazaAay' is shown at the bottom.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 2: Reverse Bits

Code: -

```
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;  
    }  
};
```

Submission: -

The screenshot displays a code submission interface for a C++ problem. The left sidebar shows the submission status as 'Accepted' with 600/600 testcases passed. The runtime is 4 ms, beating 32.46% of other solutions. The memory usage is 7.58 MB, beating 98.98% of other solutions. A bar chart shows the runtime distribution across different time intervals. The main area displays the C++ code for the 'reverseBits' function. The right sidebar shows the test results, indicating that the code was accepted with a runtime of 0 ms. The input and output for the test case are also shown.

**Runtime Statistics:**

Time Interval	Percentage
0ms - 1ms	~55%
1ms - 2ms	~5%
2ms - 3ms	~10%
3ms - 4ms	~5%
4ms - 5ms	~25%

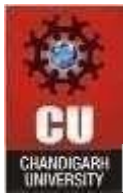
**Test Results:**

Accepted Runtime: 0 ms

Case 1 Case 2

Input: n = 000000101001010000001111010011100

Output: 964176192 (00111001011110000010100101000000)



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 3: Number of 1 Bits

Code: -

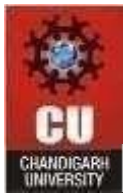
```
class Solution {  
public:  
    int hammingWeight(int n) {  
        int count=0;  
        while(n!=0){  
            if(n&1){  
                count++;  
            }  
            n=n>>1;  
        }  
        return count;  
    }  
};
```

Submission: -

The screenshot displays a code submission interface for a problem titled "Number of 1 Bits". The interface is divided into several sections:

- Problem List:** Shows the problem name and status (Accepted).
- Description:** Indicates that 598 / 598 testcases passed.
- Editorial:** A button to view the editorial.
- Solution:** A button to view the solution.
- Runtime:** Shows 0 ms and 100.00% beats.
- Memory:** Shows 8.32 MB and 12.24% beats.
- Code:** A C++ code editor showing the following code:

```
int count=0;  
while(n!=0){  
    if(n&1){  
        count++;  
    }  
    n=n>>1;  
}  
return count;
```
- Testcase:** A section for testing the code with a specific input. The input is 11.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 4: Maximum Subarray

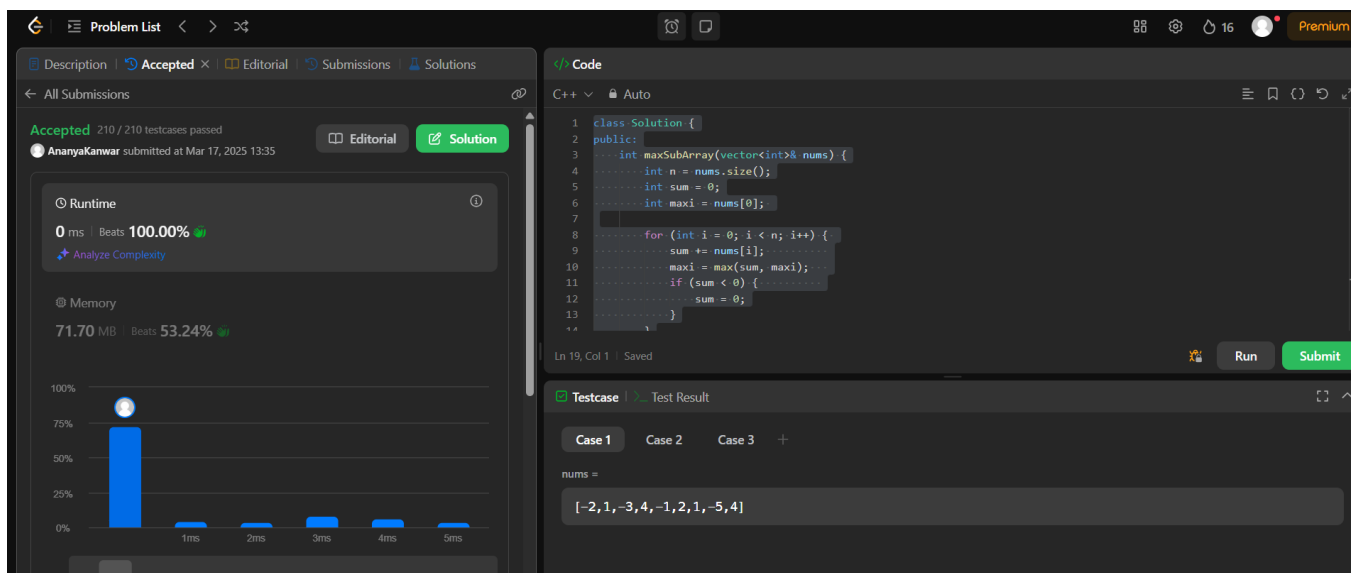
Code: -

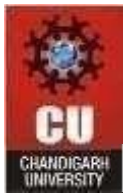
```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int n = nums.size();
        int sum = 0;
        int maxi = nums[0];

        for (int i = 0; i < n; i++) {
            sum += nums[i];
            maxi = max(sum, maxi);
            if (sum < 0) {
                sum = 0;
            }
        }

        return maxi;
    }
};
```

Submission: -





# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 5: Search a 2D Matrix II

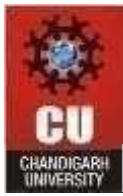
Code: -

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

Submission: -

The screenshot displays a submission page for a problem titled "Search a 2D Matrix II". The interface is divided into several sections:

- Problem List:** Shows the problem name and a "Premium" badge.
- Accepted:** Indicates that 600 out of 600 testcases passed. The submission was made by "AnanyaKanwar" on Mar 17, 2025 at 22:43.
- Runtime:** Shows a runtime of 4 ms, which beats 32.46% of other submissions.
- Memory:** Shows a memory usage of 7.58 MB, which beats 98.98% of other submissions.
- Code:** Displays the C++ code used for the solution. The code defines a class `Solution` with a public method `searchMatrix` that uses a binary search approach to find the target in the matrix.
- Test Result:** Shows the results for two test cases. Both cases are marked as "Accepted". The input for Case 1 is a 2D matrix and a target value, and the output is the result of the search.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 6: Super Pow

Code: -

```
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);
    }
};
```

Submission: -

The screenshot shows a submission on a coding platform. The left panel displays the submission status: 'Accepted' with 57/57 testcases passed. The user 'AnanyaKanwar' submitted at Mar 17, 2025 22:47. The runtime is 2 ms, beating 35.21% of solutions. The memory usage is 15.21 MB, beating 52.24% of solutions. A bar chart shows the runtime distribution across 6ms intervals. The right panel shows the C++ code for the 'superPow' function, which uses a recursive-like approach with a private 'solve' method to calculate the result modulo 1337. The code is submitted and the test result is 'Accepted' with a runtime of 0 ms. The input for the test case is 'a = 2' and 'b = [3]'.

**Submission Details:**

- Status: Accepted (57 / 57 testcases passed)
- Submitted by: AnanyaKanwar (Mar 17, 2025 22:47)
- Runtime: 2 ms (Beats 35.21%)
- Memory: 15.21 MB (Beats 52.24%)

**Code:**

```
C++
17  a%=1337;
18  int n = b.size();
19  int m = 1140;
20  int expi = 0;
21  for(int i : b){
22      expi = (expi*10+i)%m;
23  }
24  if (expi == 0) { expi = m; }
25  return solve(a,expi,1337);
26  }
```

**Test Result:**

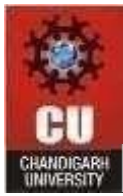
Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input:

a = 2

b = [3]



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 7: Beautiful Array

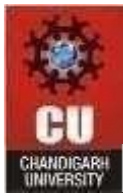
Code: -

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

Submission: -

The screenshot displays a code submission interface for a problem titled "Beautiful Array". The interface is divided into several sections:

- Problem List:** Shows the problem name and a "Solution" button.
- Accepted:** Indicates that the solution is accepted, with 38 / 38 testcases passed. The user "AnanyaKanwar" submitted it on Mar 17, 2025 at 22:50.
- Runtime:** Shows a runtime of 3 ms, which beats 38.89% of other solutions. A link to "Analyze Complexity" is provided.
- Memory:** Shows a memory usage of 10.12 MB, which beats 40.60% of other solutions.
- Code:** Displays the C++ code for the solution, which is the same as the code provided in the previous block. The code is in C++ and uses a vector to build the beautiful array.
- Testcase:** Shows the test result for the first case, which is "Accepted" with a runtime of 0 ms.

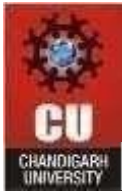


## Ques 8: The Skyline Problem

Code: -

```
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
        int edge_idx = 0; vector<pair<int, int>> edges;
        priority_queue<pair<int, int>> pq;
        vector<vector<int>> skyline;
        for (int i = 0; i < buildings.size(); ++i) {
            const auto &b = buildings[i]; edges.emplace_back(b[0], i);
            edges.emplace_back(b[1], i); }
        std::sort(edges.begin(), edges.end());
        while (edge_idx < edges.size()) {
            int curr_height;
            const auto &[curr_x, _] = edges[edge_idx];
            while (edge_idx < edges.size() &&
                    curr_x == edges[edge_idx].first) {
                const auto &[, building_idx] = edges[edge_idx];
                const auto &b = buildings[building_idx];
                if (b[0] == curr_x)
                    pq.emplace(b[2], b[1]);
                ++edge_idx; }
            while (!pq.empty() && pq.top().second <= curr_x)
                pq.pop();
            curr_height = pq.empty() ? 0 : pq.top().first;
            if (skyline.empty() || skyline.back()[1] != curr_height)
                skyline.push_back({curr_x, curr_height}); }
        return skyline;
    }
};
```





# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Submission: -

DescriptionEditorialSubmissionsSolutionsAccepted

< All Submissions

Accepted 44 / 44 testcases passed

AnanyaKanwar submitted at Mar 17, 2025 22:53

EditorialSolution

Runtime

13 ms | Beats 72.52%

Analyze Complexity

Memory

26.60 MB | Beats 92.24%

Code

C++Auto

```
24         if (b[0] == curr_x)
25             pq.emplace(b[2], b[1]);
26             ++edge_idx;
27         }
28         while (!pq.empty() && pq.top().second <= curr_x)
29             pq.pop();
30         curr_height = pq.empty() ? 0 : pq.top().first;
31         if (skyline.empty() || skyline.back()[1] != curr_height)
32             skyline.push_back({curr_x, curr_height});
33     }
34     return skyline;
35 }
36 
```

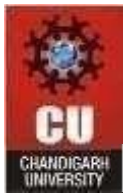
Ln 33, Col 10 / Saved

RunSubmit

TestcaseTest Result

Accepted Runtime: 0 ms

Case 1Case 2



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

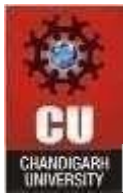
## Ques 9: Reverse Pairs

Code: -

```
class Solution {
private:

int countPairs(vector<int>& arr,int low,int mid,int high){
    int cnt=0;
    int right=mid+1;
    for(int i=low;i<=mid;i++){
        while(right<=high && 0.5*arr[i]>arr[right]) right++;
        cnt+=right-(mid+1);
    }
    return cnt;
}

void merge(vector<int>& arr,int low,int mid,int high){
    int left=low;
    int right=mid+1;
    vector<int> temp;
    while(left<=mid && right<=high){
        if(arr[left]<=arr[right]){
            temp.push_back(arr[left]);
            left++;
        }
        else{
            temp.push_back(arr[right]);
            right++;
        }
    }
    while(left<=mid){
        temp.push_back(arr[left]);
        left++;
    }
    while(right<=high){
        temp.push_back(arr[right]);
    }
}
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

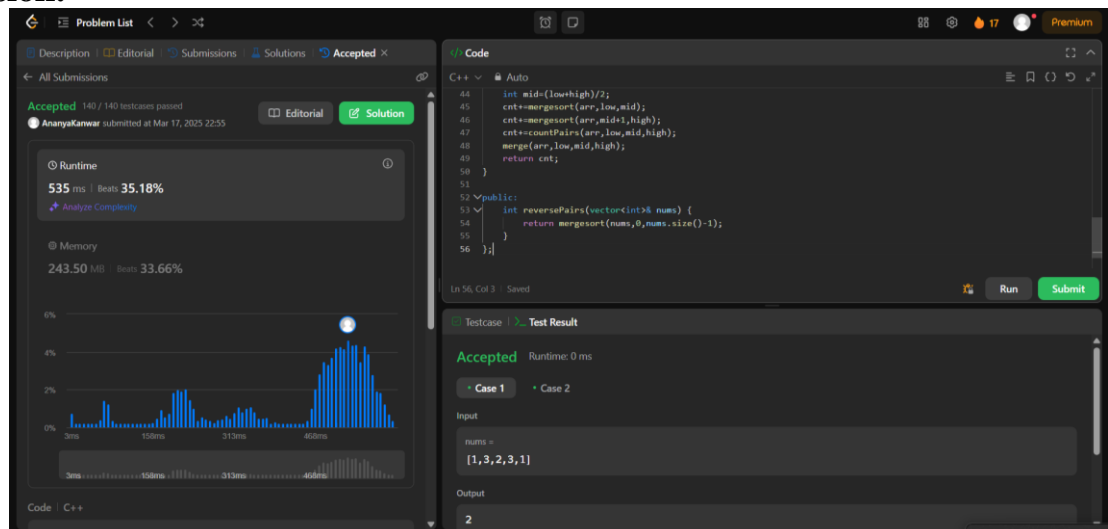
Discover. Learn. Empower.

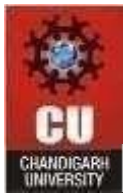
```
        right++;
    }
    for(int i=low;i<=high;i++){
        arr[i]=temp[i-low];
    }
}

int mergesort(vector<int>& arr,int low,int high){
    int cnt=0;
    if(low>=high) return cnt;
    int mid=(low+high)/2;
    cnt+=mergesort(arr,low,mid);
    cnt+=mergesort(arr,mid+1,high);
    cnt+=countPairs(arr,low,mid,high);
    merge(arr,low,mid,high);
    return cnt;
}

public:
    int reversePairs(vector<int>& nums) {
        return mergesort(nums,0,nums.size()-1);
    }
};
```

**Submission: -**





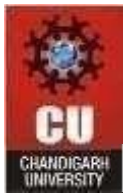
# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Ques 10: Longest Increasing Subsequence II

Code: -

```
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--){
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
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];  
}  
};
```

Submission: -

The screenshot displays a submission interface for a C++ problem. The left sidebar shows the problem list and submission details. The main area is divided into three sections: a code editor, a runtime/memory analysis section, and a test result section.

**Code Editor:** The code is written in C++ and implements a segment tree for range updates and point queries. The code is as follows:

```
29 if(n==1) return 1;  
30 int m=*max_element(nums.begin(),nums.end());  
31 tree.clear();  
32 tree.resize(4*m+10);  
33 for(int i=n-1;i>=0;i--){  
34     int l=nums[i]+1,r=min(nums[i]+k,m);  
35     int x=query(1,0,m,l,r);  
36     if(x==-1e9) x=0;  
37     update(1,0,m,nums[i],x+1);  
38 }  
39 return tree[1];  
40 }  
41 };
```

**Runtime/Memory Analysis:** The submission is marked as "Accepted" with 84/84 testcases passed. The runtime is 67 ms, which beats 81.37% of other submissions. The memory usage is 59.86 MB, which beats 77.02% of other submissions. A bar chart shows the distribution of runtime across different test cases.

**Test Result:** The submission is marked as "Accepted" with a runtime of 0 ms. The input for the test case is:

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