Q1. The King's Feast The King has n plates of food, each with a certain quantity. He wants to know the maximum food plate.

- Input: n=5, arr=[2,7,1,9,5]
- Output: 9
- Constraints: 1 ≤ n ≤ 10^5, -10^9 ≤ arr[i] ≤ 10^9

Home > test 35 > kingsfeast.cpp > main()

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int N;
    cin>>N;
    int arr[N],maxi=INT_MIN;
    for(int i=0; i<N; i++){
        cin>>arr[i];
        maxi=max(arr[i],maxi);
    }
    cout<<"Max element is: "<<maxi<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
5\" ; if ($?) { g++ kingsfeast.cpp -o kingsfeast } ; if ($?) { .\kingsfeast
 }
5
1
2
3
4
5
Max element is: 5
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\test 35>
```

Q2. The Lost Soldier In the battlefield, soldiers are numbered 0…n. One soldier is missing. Find him.

- Input: n=5, arr=[0,1,2,4,5]
- Output: 3
- Constraints: O(n) or O(log n) solution required.

Home > test 35 > lostsoldier.cpp > main()

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=11;
    int arr[]={0,1,2,3,4,5,6,7,8,10,11};
    int st=0;
    int end=n-1;
    int mid=st+(end-st)/2;
    while(st<end){
        mid=st+(end-st)/2;
        if(mid==arr[mid]){
            st=mid+1;
        }else{
            end=mid;
        }
    }cout << "The lost soldier is: " << st << endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive -
stsoldier } ; if ($?) { .\lostsoldier }
The lost soldier is: 9
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\test 35>
```

Q3. Potion Mixing (Two Sum) A wizard wants to mix two potions whose strengths add up to target.

- Input: n=4, arr=[3,2,4,7], target=6
- Output: Indices (1,2)
- Constraints: 1 ≤ n ≤ 10^5, -10^9 ≤ arr[i] ≤ 10^9

kingsfeast.cpp | lostsoldier.cpp | twosum.cpp

Home > test 35 > twosum.cpp > ...

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    int target;
    cin>>target;
    unordered_map<int,int> m;
    int ans1=-1,ans2=-1;
    for(int i=0;i<n;i++){
        int need=target-arr[i];
        if(m.find(need)!=m.end()){
            ans1=m[need]+1;
            ans2=i+1;
            break;
        }
        m[arr[i]]=i;
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
pp -o twosum } ; if ($?) { .\twosum }
5
1
2
3
4
5
5
Indices (2,3)
```

Q4. The Secret Message A spy wrote a secret message as numbers. To decode it, reverse the array.

- Input: arr=[1,2,3,4]
- Output: [4,3,2,1]

Home > test 35 > reverse.cpp > main()

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    int st=0;
    int end=n-1;
    while(st<=end){
        swap(arr[st],arr[end]);
        st++;
        end--;
    }
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
cpp -o reverse } ; if ($?) { .\reverse }
5
1
2
3
4
5
5 4 3 2 1
```

Q5. The King's Parade Soldiers stand in line. Check if their heights are sorted in non-decreasing order.

- Input: arr=[1,3,5,7] → Output: true
- Input: arr=[3,2,1] → Output: false

Home > test 35 > sortedhei.cpp > ...

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    bool sorted=true;
    for(int i=1;i<n;i++){
        if(arr[i]<arr[i-1]){
            sorted=false;
            break;
        }
    }
    if(sorted==true){
        cout<<"true"<<endl;
    }else{
        cout<<"false"<<endl;
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
i.cpp -o sortedhei } ; if ($?) { .\sortedhei }
5
1
2
3
4
8
true
```

Q6. The Treasure Island Each island grid has gold. Find the island row with maximum gold.

- Input: 3 3 1 2 3 4 5 6 7 8 9
- Output: Row 2 (sum=24)

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n,m;
    cin>>n>>m;
    int arr[n][m];
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            cin>>arr[i][j];
        }
    }
    int maxi=INT_MIN;
    int row=-1;
    for(int i=0;i<n;i++){
        int sum=0;
        for(int j=0;j<m;j++){
            sum+=arr[i][j];
        }
        if(sum>maxi){
            maxi=sum;
            row=i+1;
        }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd
edhei } ; if ($?) { .\sortedhei }
3
1
2
3
5
Row 3 (sum=5)
```

Q7. The Spiral Library The King built a library where books are kept in spiral shelves. Print them in spiral order.

- Input: 3 3 1 2 3 4 5 6 7 8 9
- Output: [1,2,3,6,9,8,7,4,5]

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n,m;
    cin>>n>>m;
    int arr[n][m];
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            cin>>arr[i][j];
        }
    }
    vector<int> ans;
    int top=0,bottom=n-1,left=0,right=m-1;
    while(top<=bottom && left<=right){
        for(int i=left;i<=right;i++){
            ans.push_back(arr[top][i]);
        }
        top++;
        for(int i=top;i<=bottom;i++){
            ans.push_back(arr[i][right]);
        }
        right--;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
3
3
1
2
3
4
5
6
7
8
9
1 2 3 6 9 8 7 4 5
```

Q8. The Royal Diagonal In a royal hall represented as a square, find sum of both diagonals.

Input: 3 3 1 2 3 4 5 6 7 8 9

Output: 1+5+9 =15, 3+5+7 = 15

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n,m;
    cin>>n>>m;
    int arr[n][m];
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            cin>>arr[i][j];
        }
    }
    int sum1=0,sum2=0;
    for(int i=0;i<n;i++){
        sum1+=arr[i][i];
        sum2+=arr[i][m-i-1];
    }
    cout<<"Primary Diagonal Sum = "<<sum1<<", Secondary Diagonal Sum = "<<sum2<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
3
3
1
2
3
4
5
6
7
8
9
Primary Diagonal Sum = 15, Secondary Diagonal Sum = 15
```

Q9. The Messenger's Path A messenger wants to go from (0,0) to (n-1,m-1). Cells with 1 are blocked. Can he reach?
Input: 3 3 0 0 0 0 1 0 0 0 0
Output: true

```cpp
#include<bits/stdc++.h>
using namespace std;

int n=3,m=3;
int arr[3][3]={{0,0,0},{0,1,0},{0,0,0}};
int vis[3][3]={0};
int a[4]={1,-1,0,0};
int b[4]={0,0,1,-1};

bool dfs(int x,int y){
    if(x==n-1 && y==m-1) return true;
    vis[x][y]=1;
    for(int i=0;i<4;i++){
        int nx=x+a[i];
        int ny=y+b[i];
        if(nx>=0 && ny>=0 && nx<n && ny<m && arr[nx][ny]==0 && vis[nx][ny]==0){
            if(dfs(nx,ny)) return true;
        }
    }
    return false;
}

int main(){
    if(dfs(0,0)){
        cout<<"true"<<endl;
    }else{
        cout<<"false"<<endl;
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
> cd "d:\OneDrive - gai
i.cpp -o sortedhei } ; if ($?) { .\sortedhei }
true
```

Q10. The Rainwater Pond Count the number of water ponds in a village (1 = water, 0 = land).

Input: 3 3 1 0 1 0 1 0 1 0 1 ●
Output: 5

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{1,0,1},{0,1,0},{1,0,1}};
    int count=0;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(arr[i][j]==1){
                count++;
            }
        }
    }
    cout<<count<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
1
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\
i.cpp -o sortedhei } ; if ($?) { .\sortedhei }
5
```

Q11. Tower of Temples (Hanoi) Temples have n golden disks. Move them from source → destination using helper temple. Return moves.

- Input: n=3
- Output: 7

Home > test 35 > sortedhei.cpp > ...

```cpp
int hanoi(int n){
    if(n==1) return 1;
    return 2*hanoi(n-1)+1;
}

int main(){
    int n=3;
    int moves=hanoi(n);
    cout<<moves<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\

i.cpp -o sortedhei } ; if ($?) { .\sortedhei }
7
```

Q12. The Magical Staircase A child climbs 1 or 2 steps. Find number of ways to reach step n.

- Input: n=4
- Output: 5

```cpp
#include<bits/stdc++.h>
using namespace std;

int stairs(int n){
    if(n==0 || n==1) return 1;
    return stairs(n-1)+stairs(n-2);
}

int main(){
    int n=4;
    int ways=stairs(n);
    cout<<ways<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackat
ortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
5
```

Q13. The Sorcerer's Spell Reverse a string using recursion.

- Input: abc
- Output: cba

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;

void reverseStr(char s[], int l, int r){
    if(l>=r) return;
    swap(s[l], s[r]);
    reverseStr(s, l+1, r-1);
}

int main(){
    char s[]="abc";
    int n=strlen(s);
    reverseStr(s,0,n-1);
    for(int i=0;i<n;i++){
        cout<<s[i];
    }
    cout<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive - 
ortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
cba
```

Q14. The Dragon's Roar Print numbers 1 to n using recursion.

- Input: n=5
- Output: 1 2 3 4 5

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;

void printNum(int n){
    if(n==0) return;
    printNum(n-1);
    cout<<n<<" ";
}

int main(){
    int n=5;
    printNum(n);
    cout<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\

 ($?) { g++ sortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
1 2 3 4 5
```

Q15. The Hidden Chamber Find sum of array elements using recursion.

- Input: arr=[1,2,3,4]
- Output: 10

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;

int sumArr(int arr[], int n){
    if(n==0) return 0;
    return arr[n-1]+sumArr(arr,n-1);
}

int main(){
    int arr[]={1,2,3,4};
    int n=4;
    int sum=sumArr(arr,n);
    cout<<sum<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive -
ortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
10
```

Q16. The Ancient Scroll Search for a scroll ID in the archive.

- Input: arr=[2,5,7,8], key=7
- Output: 2

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={2,5,7,8};
    int n=4;
    int key=7;
    int ans=-1;
    for(int i=0;i<n;i++){
        if(arr[i]==key){
            ans=i+1;
            break;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\tes

 ($?) { g++ sortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
3
```

Q17. The Farmer's Basket Find if a fruit (number) exists in the basket.

- Input: arr=[10,20,30], key=25
- Output: -1

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={10,20,30};
    int n=3;
    int key=25;
    int ans=-1;
    for(int i=0;i<n;i++){
        if(arr[i]==key){
            ans=i+1;
            break;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneD
ortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
-1
```

Q18. The Secret Door Doors are numbered in increasing order. Find target door using binary search.

- Input: arr=[1,3,5,7,9], key=7
- Output: 3

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,3,5,7,9};
    int n=5;
    int key=7;
    int st=0,end=n-1,ans=-1;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]==key){
            ans=mid+1;
            break;
        }else if(arr[mid]<key){
            st=mid+1;
        }else{
            end=mid-1;
        }
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\t

 ($?) { g++ sortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
4
```

Q19. The Archer's Range Find the first occurrence of an arrow's distance.

- Input: arr=[1,2,2,2,3], key=2
- Output: 1

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,2,2,3};
    int n=5;
    int key=2;
    int st=0,end=n-1,ans=-1;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]==key){
            ans=mid+1;
            end=mid-1;
        }else if(arr[mid]<key){
            st=mid+1;
        }else{
            end=mid-1;
        }
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon>
ortedhei.cpp -o sortedhei } ; if ($?) { .\sortedhei }
2
```

Q20. The Treasure Chest Find the last occurrence of a key using binary search.

- Input: arr=[1,2,2,2,3], key=2
- Output: 3

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,2,2,3};
    int n=5;
    int key=2;
    int st=0,end=n-1,ans=-1;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]==key){
            ans=mid+1;
            st=mid+1;
        }else if(arr[mid]<key){
            st=mid+1;
        }else{
            end=mid-1;
        }
    }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hacka

 ($?) { g++ sortedhei.cpp -o sortedhei } ; if ($?) { .\so
4
```

Q21.The first index where the element is greater than or equal to the target.

- If element is found → return its first occurrence.
- If not found → return position where it can be inserted.
- If not possible → return n (array size). Example Array = [1, 2, 4, 6, 6, 8], target = 6

- Lower bound = index 3 (first 6). Array = [1, 2, 4, 6, 6, 8], target = 5
- Lower bound = index 3 (as 6 is the first ≥ 5).

Home > test 35 > sortedhei.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,4,6,6,8};
    int n=6;
    int target=6;
    int st=0,end=n-1,ans=n;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]>=target){
            ans=mid;
            end=mid-1;
        }else{
            st=mid+1;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackath
 ($?) { g++ sortedhei.cpp -o sortedhei } ; if ($?) { .\sort
3
```

Q22.The first index where the element is strictly greater than the target.

- If all elements ≤ target → return n. Example Array = [1, 2, 4, 6, 6, 8], target = 6
- Upper bound = index 5 (first element greater than 6 is 8). Array = [1, 2, 4, 6, 6, 8], target = 7
- Upper bound = index 5 (8 is first > 7).

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,4,6,6,8};
    int n=6;
    int target=6;
    int st=0,end=n-1,ans=n;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]>target){
            ans=mid+1;
            end=mid-1;
        }else{
            st=mid+1;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon
wosum.cpp -o twosum } ; if ($?) { .\twosum }
6
```

Q23.The smallest element ≥ target (actual value, not index).

- If no such element exists → return -1. Example Array = [1, 2, 4, 6, 6, 8], target = 5
- Ceil = 6. Array = [1, 2, 4, 6, 6, 8], target = 9
- Ceil = -1 (no element ≥ 9).

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,4,6,6,8};
    int n=6;
    int target=5;
    int st=0,end=n-1,ans=-1;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]>=target){
            ans=arr[mid];
            end=mid-1;
        }else{
            st=mid+1;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\te

 ($?) { g++ twosum.cpp -o twosum } ; if ($?) { .\twosum }
6
```

Q24. The largest element ≤ target.

- If no such element exists → return -1. Example Array = [1, 2, 4, 6, 6, 8], target = 5
- Floor = 4. Array = [1, 2, 4, 6, 6, 8], target = 0
- Floor = -1 (no element ≤ 0).

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int arr[]={1,2,4,6,6,8};
    int n=6;
    int target=5;
    int st=0,end=n-1,ans=-1;
    while(st<=end){
        int mid=st+(end-st)/2;
        if(arr[mid]<=target){
            ans=arr[mid];
            st=mid+1;
        }else{
            end=mid-1;
        }
    }
    cout<<ans<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop
wosum.cpp -o twosum } ; if ($?) { .\twosum }
4
```

Q25. The Treasure Map (Linear Search) A treasure map is represented as a grid n x m. Each cell contains a number. The

King wants to know if the treasure (target) exists on the map.

- Input: n=3, m=3 matrix = [[1,2,3], [4,5,6], [7,8,9]] target = 5
- Output: Yes
- Constraints: 1 ≤ n,m ≤ 500, -10^6 ≤ matrix[i][j] ≤ 10^6

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{1,2,3},{4,5,6},{7,8,9}};
    int target=5;
    int found=0;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(arr[i][j]==target){
                found=1;
                break;
            }
        }
    }
    if(found==1){
        cout<<"Yes"<<endl;
    }else{
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\test 3

 ($?) { g++ twosum.cpp -o twosum } ; if ($?) { .\twosum }
Yes
```

Q26. The Magical Scrolls (Linear Search Return Index) In the royal library, scrolls are arranged in a 2D cabinet of size n x m.

Find the row and column of the scroll with ID = target. If not found, return (-1,-1).

- Input: matrix = [[10,20,30], [40,50,60], [70,80,90]] target = 60
- Output: (1,2)
- Constraints: 1 ≤ n,m ≤ 1000

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{10,20,30},{40,50,60},{70,80,90}};
    int target=60;
    int row=-1,col=-1;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(arr[i][j]==target){
                row=i+1;
                col=j+1;
                break;
            }
        }
        if(row!=-1) break;
    }
    cout<<"("<<row<<","<<col<<")"<<endl;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\test 35>
> cd "
($?) { g++ twosum.cpp -o twosum } ; if ($?) { .\twosum }
(2,3)
```

Q27. The Battle Formation (Binary Search - Flattened)
Soldiers stand in a grid formation. Their strengths are sorted row-wise and the first element of each row is greater than the last of the previous row. The commander wants to know if a soldier with strength x exists.

- Input: matrix = [[1,3,5], [7,10,11], [16,20,30]] target = 10
- Output: True
- Constraints: $1 \leq n,m \leq 300$

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{1,3,5},{7,10,11},{16,20,30}};
    int target=10;
    int row=-1;
    for(int i=0;i<n;i++){
        if(arr[i][0]<=target && arr[i][m-1]>=target){
            row=i;
            break;
        }
    }
    int found=0;
    if(row!=-1){
        int st=0,end=m-1;
        while(st<=end){
            int mid=st+(end-st)/2;
            if(arr[row][mid]==target){
                found=1;
                break;
            }else if(arr[row][mid]<target){
                st=mid+1;
            }else{
                end=mid-1;
            }
        }
    }
    if(found==1){
        cout<<"True"<<endl;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> c
wosum.cpp -o twosum } ; if ($?) { .\twosum }
True
```

Q28. The Queen's Jewels (Binary Search First Occurrence)
The Queen's jewels are stored in a 2D sorted grid. She wants to find the first position of a jewel type x.

- Input: matrix = [[1,2,2], [3,4,4], [5,6,7]] target = 4
- Output: (1,1)
- Constraints: $1 \leq n,m \leq 1000$

Home > test 35 > twosum.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{1,2,2},{3,4,4},{5,6,7}};
    int target=4;
    int row=-1,col=-1;
    for(int i=0;i<n;i++){
        if(arr[i][0]<=target && arr[i][m-1]>=target){
            row=i;
            break;
        }
    }
    if(row!=-1){
        int st=0,end=m-1;
        while(st<=end){
            int mid=st+(end-st)/2;
            if(arr[row][mid]==target){
                col=mid+1;
                break;
            }else if(arr[row][mid]<target){
                st=mid+1;
            }else{
                end=mid-1;
            }
        }
    }
    cout<<"("<<row+1<<","<<col<<")"<<endl;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
> cd "d:\OneDrive
($?) { g++ twosum.cpp -o twosum } ; if ($?) { .\twosum }
(2,2)
```

Q29. The Hidden Scrolls (Staircase Search) The King hides scrolls in a 2D matrix where rows and columns are sorted. Find if a scroll with ID x exists. Use O(n+m) method (start from top-right corner).

● Input: matrix = [[1,4,7,11], [2,5,8,12], [3,6,9,16], [10,13,14,17]] target = 6
● Output: True
● Constraints: 1 ≤ n,m ≤ 1000

Home > test 35 > twosum.cpp

```cpp
int main(){
    int n=4,m=4;
    int arr[4][4]={{1,4,7,11},{2,5,8,12},{3,6,9,16},{10,13,14,17}};
    int target=6;
    int i=0,j=m-1;
    int found=0;
    while(i<n && j>=0){
        if(arr[i][j]==target){
            found=1;
            break;
        }else if(arr[i][j]>target){
            j--;
        }else{
            i++;
        }
    }
    if(found==1){
        cout<<"True"<<endl;
    }else{
        cout<<"False"<<endl;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive - gal
wosum.cpp -o twosum } ; if ($?) { .\twosum }
True
```

Q30. The Magic Portal (Binary Search 2D) A wizard created portals in a 2D grid sorted in ascending order row-wise and column-wise. To activate a portal, he must find a specific number x. Return "Activated" if found else "Failed".
● Input: matrix = [[1, 2, 8], [3, 6, 10], [7, 9, 12]] target = 9

- Output: Activated
- Constraints: 1 ≤ n,m ≤ 500

kingsfeast.cpp | lostsoldier.cpp | twosum.cpp | reverse.cpp | sortedhei.cpp | whatsapp.py

Home > test 35 > twosum.cpp

```cpp
using namespace std;
int main(){
    int n=3,m=3;
    int arr[3][3]={{1,2,8},{3,6,10},{7,9,12}};
    int target=9;
    int row=-1,found=0;
    for(int i=0;i<n;i++){
        if(arr[i][0]<=target && arr[i][m-1]>=target){
            row=i;
            break;
        }
    }
    if(row!=-1){
        int st=0,end=m-1;
        while(st<=end){
            int mid=st+(end-st)/2;
            if(arr[row][mid]==target){
                found=1;
                break;
            }else if(arr[row][mid]<target){
                st=mid+1;
            }else{
                end=mid-1;
            }
        }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive - galgotiasuniversity.edu.in\Desk
wosum.cpp -o twosum } ; if ($?) { .\twosum }
True
```