

# TEST - 35 (SAAD KHAN <SEC-34>):-

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:  $1 \leq n \leq 10^5$ ,  $-10^9 \leq \text{arr}[i] \leq 10^9$



```
Home > test 35 > kingsfeast.cpp > main()
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int N;
5      cin>>N;
6      int arr[N],maxi=INT_MIN;
7      for(int i=0; i<N; i++){
8          cin>>arr[i];
9          maxi=max(arr[i],maxi);
10     }
11     cout<<"Max element is: "<<maxi<<endl;
12 }
```

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()

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=11;
5      int arr[]={0,1,2,3,4,5,6,7,8,10,11};
6      int st=0;
7      int end=n-1;
8      int mid=st+(end-st)/2;
9      while(st<end){
10         mid=st+(end-st)/2;
11         if(mid==arr[mid]){
12             st=mid+1;
13         }else{
14             end=mid;
15         }
16     }cout << "The lost soldier is: " << st << endl;
17 }
```

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:  $1 \leq n \leq 10^5$ ,  $-10^9 \leq arr[i] \leq 10^9$

```
kingsfeast.cpp  lostsoldier.cpp  twosum.cpp X
Home > test 35 > twosum.cpp > ...
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n;
5      cin>>n;
6      int arr[n];
7      for(int i=0;i<n;i++){
8          cin>>arr[i];
9      }
10     int target;
11     cin>>target;
12     unordered_map<int,int> m;
13     int ans1=-1,ans2=-1;
14     for(int i=0;i<n;i++){
15         int need=target-arr[i];
16         if(m.find(need)!=m.end()){
17             ans1=m[need]+1;
18             ans2=i+1;
19             break;
20         }
21         m[arr[i]]=i;
22     }

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()
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n;
5      cin>>n;
6      int arr[n];
7      for(int i=0;i<n;i++){
8          cin>>arr[i];
9      }
10     int st=0;
11     int end=n-1;
12     while(st<=end){
13         swap(arr[st],arr[end]);
14         st++;
15         end--;
16     }
17     for(int i=0;i<n;i++){
18         cout<<arr[i]<<" ";
19     }
20 }
```

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 > ...

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n;
5      cin>>n;
6      int arr[n];
7      for(int i=0;i<n;i++){
8          cin>>arr[i];
9      }
10     bool sorted=true;
11     for(int i=1;i<n;i++){
12         if(arr[i]<arr[i-1]){
13             sorted=false;
14             break;
15         }
16     }
17     if(sorted==true){
18         cout<<"true"<<endl;
19     }else{
20         cout<<"false"<<endl;
21     }
22 }
```

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)

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n,m;
5      cin>>n>>m;
6      int arr[n][m];
7      for(int i=0;i<n;i++){
8          for(int j=0;j<m;j++){
9              cin>>arr[i][j];
10         }
11     }
12     int maxi=INT_MIN;
13     int row=-1;
14     for(int i=0;i<n;i++){
15         int sum=0;
16         for(int j=0;j<m;j++){
17             sum+=arr[i][j];
18         }
19         if(sum>maxi){
20             maxi=sum;
21             row=i+1;
22     }
```

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]

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n,m;
5      cin>>n>>m;
6      int arr[n][m];
7      for(int i=0;i<n;i++){
8          for(int j=0;j<m;j++){
9              cin>>arr[i][j];
10         }
11     }
12     vector<int> ans;
13     int top=0,bottom=n-1,left=0,right=m-1;
14     while(top<=bottom && left<=right){
15         for(int i=left;i<=right;i++){
16             ans.push_back(arr[top][i]);
17         }
18         top++;
19         for(int i=top;i<=bottom;i++){
20             ans.push_back(arr[i][right]);
21         }
22         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:  $1+5+9=15$ ,  $3+5+7=15$

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n,m;
5      cin>>n>>m;
6      int arr[n][m];
7      for(int i=0;i<n;i++){
8          for(int j=0;j<m;j++){
9              cin>>arr[i][j];
10         }
11     }
12     int sum1=0,sum2=0;
13     for(int i=0;i<n;i++){
14         sum1+=arr[i][i];
15         sum2+=arr[i][m-i-1];
16     }
17     cout<<"Primary Diagonal Sum = "<<sum1<<" , Secondary Diagonal Sum = "<<sum2<<endl;
18 }
19
```

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

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int n=3,m=3;
5  int arr[3][3]={0,0,0},{0,1,0},{0,0,0};
6  int vis[3][3]={0};
7  int a[4]={1,-1,0,0};
8  int b[4]={0,0,1,-1};
9
10 bool dfs(int x,int y){
11     if(x==n-1 && y==m-1) return true;
12     vis[x][y]=1;
13     for(int i=0;i<4;i++){
14         int nx=x+a[i];
15         int ny=y+b[i];
16         if(nx>=0 && ny>=0 && nx<n && ny<m && arr[nx][ny]==0 && vis[nx][ny]==0){
17             if(dfs(nx,ny)) return true;
18         }
19     }
20     return false;
21 }
22
23 int main(){
24     if(dfs(0,0)){
25         cout<<"true"<<endl;
26     }else{
27         cout<<"false"<<endl;
28     }
29 }
30
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
> cd "d:\OneDrive - ga
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 •

Output: 5

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={{1,0,1},{0,1,0},{1,0,1}};
6      int count=0;
7      for(int i=0;i<n;i++){
8          for(int j=0;j<m;j++){
9              if(arr[i][j]==1){
10                 count++;
11             }
12         }
13     }
14     cout<<count<<endl;
15 }
16
```

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 > ...  
4  int hanoi(int n){  
5      if(n==1) return 1;  
6      return 2*hanoi(n-1)+1;  
7  }  
8  
9  int main(){  
10     int n=3;  
11     int moves=hanoi(n);  
12     cout<<moves<<endl;  
13 }  
14  
  
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

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int stairs(int n){
5      if(n==0 || n==1) return 1;
6      return stairs(n-1)+stairs(n-2);
7  }
8
9  int main(){
10     int n=4;
11     int ways=stairs(n);
12     cout<<ways<<endl;
13 }
14

```

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
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  void reverseStr(char s[], int l, int r){
5      if(l>=r) return;
6      swap(s[l], s[r]);
7      reverseStr(s, l+1, r-1);
8  }
9
10 int main(){
11     char s[]="abc";
12     int n=strlen(s);
13     reverseStr(s,0,n-1);
14     for(int i=0;i<n;i++){
15         cout<<s[i];
16     }
17     cout<<endl;
18 }
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon" & gcc sortedhei.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
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  void printNum(int n){
5      if(n==0) return;
6      printNum(n-1);
7      cout<<n<<" ";
8  }
9
10 int main(){
11     int n=5;
12     printNum(n);
13     cout<<endl;
14 }
15

```

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 > G+ sortedhei.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int sumArr(int arr[], int n){
5      if(n==0) return 0;
6      return arr[n-1]+sumArr(arr,n-1);
7  }
8
9  int main(){
10     int arr[]={1,2,3,4};
11     int n=4;
12     int sum=sumArr(arr,n);
13     cout<<sum<<endl;
14 }
15
```

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

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={2,5,7,8};
5      int n=4;
6      int key=7;
7      int ans=-1;
8      for(int i=0;i<n;i++){
9          if(arr[i]==key){
10             ans=i+1;
11             break;
12         }
13     }
14     cout<<ans<<endl;
15 }
16
```

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
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={10,20,30};
5      int n=3;
6      int key=25;
7      int ans=-1;
8      for(int i=0;i<n;i++){
9          if(arr[i]==key){
10             ans=i+1;
11             break;
12         }
13     }
14     cout<<ans<<endl;
15 }
16

```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

```

PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon> cd "d:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon"
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
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,3,5,7,9};
5      int n=5;
6      int key=7;
7      int st=0,end=n-1,ans=-1;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]==key){
11             ans=mid+1;
12             break;
13         }else if(arr[mid]<key){
14             st=mid+1;
15         }else{
16             end=mid-1;
17         }
18     }

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 > G sortedhei.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,2,2,3};
5      int n=5;
6      int key=2;
7      int st=0,end=n-1,ans=-1;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]==key){
11             ans=mid+1;
12             end=mid-1;
13         }else if(arr[mid]<key){
14             st=mid+1;
15         }else{
16             end=mid-1;
17         }
18     }
```

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

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,2,2,3};
5      int n=5;
6      int key=2;
7      int st=0,end=n-1,ans=-1;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]==key){
11             ans=mid+1;
12             st=mid+1;
13         }else if(arr[mid]<key){
14             st=mid+1;
15         }else{
16             end=mid-1;
17         }
18     }

```

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  $\geq 5$ ).

Home > test 35 >  sortedhei.cpp

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,4,6,6,8};
5      int n=6;
6      int target=6;
7      int st=0,end=n-1,ans=n;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]>=target){
11             ans=mid;
12             end=mid-1;
13         }else{
14             st=mid+1;
15         }
16     }
17     cout<<ans<<endl;
18 }
```

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  $\leq$  target  $\rightarrow$  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

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,4,6,6,8};
5      int n=6;
6      int target=6;
7      int st=0,end=n-1,ans=n;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]>target){
11             ans=mid+1;
12             end=mid-1;
13         }else{
14             st=mid+1;
15         }
16     }
17     cout<<ans<<endl;
18 }
```

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  $\geq$  target (actual value, not index).

- If no such element exists  $\rightarrow$  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  $\geq$  9).

```
Home > test 35 > twosum.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,4,6,6,8};
5      int n=6;
6      int target=5;
7      int st=0,end=n-1,ans=-1;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]>=target){
11             ans=arr[mid];
12             end=mid-1;
13         }else{
14             st=mid+1;
15         }
16     }
17     cout<<ans<<endl;
18 }
```

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  $\leq$  target.



- If no such element exists  $\rightarrow$  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  $\leq$  0).

```

Home > test 35 > twosum.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int arr[]={1,2,4,6,6,8};
5      int n=6;
6      int target=5;
7      int st=0,end=n-1,ans=-1;
8      while(st<=end){
9          int mid=st+(end-st)/2;
10         if(arr[mid]<=target){
11             ans=arr[mid];
12             st=mid+1;
13         }else{
14             end=mid-1;
15         }
16     }
17     cout<<ans<<endl;
18 }

```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```

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

```

Q25. The Treasure Map (Linear Search) A treasure map is represented as a grid  $n \times m$ . 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:  $1 \leq n, m \leq 500$ ,  $-10^6 \leq \text{matrix}[i][j] \leq 10^6$

```
Home > test 35 > twosum.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={{1,2,3},{4,5,6},{7,8,9}};
6      int target=5;
7      int found=0;
8      for(int i=0;i<n;i++){
9          for(int j=0;j<m;j++){
10             if(arr[i][j]==target){
11                 found=1;
12                 break;
13             }
14         }
15     }
16     if(found==1){
17         cout<<"Yes"<<endl;
18     }else{
19         cout<<"No"<<endl;
20     }
21 }
```

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  $n \times m$ .

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:  $1 \leq n, m \leq 1000$

Home > test 35 >  twosum.cpp

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={10,20,30},{40,50,60},{70,80,90}};
6      int target=60;
7      int row=-1,col=-1;
8      for(int i=0;i<n;i++){
9          for(int j=0;j<m;j++){
10             if(arr[i][j]==target){
11                 row=i+1;
12                 col=j+1;
13                 break;
14             }
15         }
16         if(row!=-1) break;
17     }
18     cout<<"("<<row<<","<<col<<")"<<endl;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\OneDrive - galgotiasuniversity.edu.in\Desktop\Hackathon\Home\test 35>


(\$?) { g++ twosum.cpp -o twosum } ; if (\$?) { .\twosum }  
(2,3)

> cd "

### 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

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={{1,3,5},{7,10,11},{16,20,30}};
6      int target=10;
7      int row=-1;
8      for(int i=0;i<n;i++){
9          if(arr[i][0]<=target && arr[i][m-1]>=target){
10             row=i;
11             break;
12         }
13     }
14     int found=0;
15     if(row!=-1){
16         int st=0,end=m-1;
17         while(st<=end){
18             int mid=st+(end-st)/2;
19             if(arr[row][mid]==target){
20                 found=1;
21                 break;
22             }else if(arr[row][mid]<target){
23                 st=mid+1;
24             }else{
25                 end=mid-1;
26             }
27         }
28     }
29     if(found==1){
30         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
1  #include<bits/stdc++.h>
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={{1,2,2},{3,4,4},{5,6,7}};
6      int target=4;
7      int row=-1,col=-1;
8      for(int i=0;i<n;i++){
9          if(arr[i][0]<=target && arr[i][m-1]>=target){
10             row=i;
11             break;
12         }
13     }
14     if(row!=-1){
15         int st=0,end=m-1;
16         while(st<=end){
17             int mid=st+(end-st)/2;
18             if(arr[row][mid]==target){
19                 col=mid+1;
20                 break;
21             }else if(arr[row][mid]<target){
22                 st=mid+1;
23             }else{
24                 end=mid-1;
25             }
26         }
27     }
28     cout<<"("<<row+1<<","<<col<<")"<<endl;
29 }
```

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:  $1 \leq n, m \leq 1000$

```

Home > test 35 > G+ twosum.cpp
2  using namespace std;
3  int main(){
4      int n=4,m=4;
5      int arr[4][4]={{1,4,7,11},{2,5,8,12},{3,6,9,16},{10,13,14,17}};
6      int target=6;
7      int i=0,j=m-1;
8      int found=0;
9      while(i<n && j>=0){
10         if(arr[i][j]==target){
11             found=1;
12             break;
13         }else if(arr[i][j]>target){
14             j--;
15         }else{
16             i++;
17         }
18     }
19     if(found==1){
20         cout<<"True"<<endl;
21     }else{
22         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:  $1 \leq n, m \leq 500$

```
kingsfeast.cpp  lostsoldier.cpp  twosum.cpp X  reverse.cpp  sortedhei.cpp  whatsapp.py

Home > test 35 > twosum.cpp
2  using namespace std;
3  int main(){
4      int n=3,m=3;
5      int arr[3][3]={{1,2,8},{3,6,10},{7,9,12}};
6      int target=9;
7      int row=-1,found=0;
8      for(int i=0;i<n;i++){
9          if(arr[i][0]<=target && arr[i][m-1]>=target){
10             row=i;
11             break;
12         }
13     }
14     if(row!=-1){
15         int st=0,end=m-1;
16         while(st<=end){
17             int mid=st+(end-st)/2;
18             if(arr[row][mid]==target){
19                 found=1;
20                 break;
21             }else if(arr[row][mid]<target){
22                 st=mid+1;
23             }else{
24                 end=mid-1;
25             }
26         }
27     }
28 }
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

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