

1763.Longest Nice Substring

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        string ans="";
        for(int i=0;i<s.length();i++){
            int count=0;
            string temp="";
            vector<bool> a(26,0),b(26,0);
            for(int j=i;j<s.length();j++){
                temp.push_back(s[j]);
                if(s[j]>='A' && s[j]<='Z'){
                    if(a[s[j]-'A']==0 && b[s[j]-'A']==0)
                        count++;
                    else if(a[s[j]-'A'] && !b[s[j]-'A'])
                        count--;
                    b[s[j]-'A']=1;
                }
                else{
                    if(b[s[j]-'a']==0 && a[s[j]-'a']==0)
                        count++;
                    else if(b[s[j]-'a'] && !a[s[j]-'a'])
                        count--;
                    a[s[j]-'a']=1;
                }
            }
            if(ans.size()<temp.size() && count==0)
                ans=temp;
        }
    }
}
```

```

    }

    return ans;

}

};

```

The screenshot displays a LeetCode submission for the problem "Longest Nice Substring". The submission is accepted, with a runtime of 6 ms and memory usage of 9.17 MB. The test result shows the input "YazaAay" and the output "aAa".

Code:

```

20 count++;
21 else if(b[s[j]-'a'] && !a[s[j]-'a'])
22 count--;
23 a[s[j]-'a']=1;
24 }
25 if(ans.size()<temp.size() && count==0)
26 ans=temp;
27 }
28 return ans;
29 }
30 }
31 };

```

Test Result:

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

s = "YazaAay"

Output

"aAa"

190.Reverse Bits

```

class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t ans=0;
        for(int i=0;i<32;i++){
            if((n>>i)&1)
                ans+=(1<<(31-i));
        }
    }
}

```

```

        return ans;
    }
};

```

The screenshot displays a LeetCode submission for the 'Reverse Bits' problem. The code is written in C++ and uses a loop to reverse the bits of a 32-bit integer. The submission is marked as 'Accepted' with a runtime of 0 ms, beating 100.00% of other submissions. The memory usage is 7.90 MB, beating 29.02%. A bar chart shows the runtime distribution, with the majority of submissions falling between 1ms and 4ms. The test result for Case 1 shows the input '00000010100101000001111010011100' and the output '964176192 (00111001011110000010100101000000)'.

191. Number of 1 Bits

```

class Solution {
public:
    int hammingWeight(int n) {
        return __builtin_popcount(n);
    }
};

```

The screenshot shows a LeetCode submission for the problem "Number of 1-Bits". The submission is accepted, with a runtime of 0 ms and memory usage of 8.16 MB. The code is in C++ and uses a while loop to count the number of 1s in the binary representation of n. The test result shows Case 1 with input n=11 and output 3.

```

5 while(n!=0){
6     if(n&1==1){
7         set++;
8     }
9     n=n>>1;
10 }
11 return set;
12 }
13
14
15
16 };

```

Testcase 1: Runtime: 0 ms
Accepted
Case 1 Case 2 Case 3
Input: n = 11
Output: 3

53. Maximum Subarray

```
class Solution {
```

```
public:
```

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

```
        int curr=0;
```

```
        int max=nums[0];
```

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

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

```
            if(curr>max)
```

```
                max=curr;
```

```
            if(curr<0)
```

```
                curr=0;
```

```

    }

    return max;

}

};

```

The screenshot shows a LeetCode submission for the "Maximum Subarray" problem. The submission is accepted, with a runtime of 0 ms and memory usage of 71.86 MB. The code is in C++ and implements a single-pass algorithm. The test result shows the input array [-2, 1, -3, 4, -1, 2, 1, -5, 4] and the output 6.

```

4      int sum=0;
5      int m=nums[0];
6      for(int i=0;i<nums.size();i++){
7          sum+=nums[i];
8          m=max(m,sum);
9          if(sum<0)
10             sum=0;
11      }
12      return m;
13  }
14  };
15

```

Testcase 1: Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input: nums = [-2, 1, -3, 4, -1, 2, 1, -5, 4]

Output: 6

240. Search a 2D Matrix II

```

class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        for(int i=0;i<matrix.size();i++){
            if(matrix[i][0]<=target && matrix[i][matrix[i].size()-1]>=target){
                int start=0;
                int end=matrix[i].size()-1;

```

```

while(start<=end){
    int mid=start+(end-start)/2;

    if(matrix[i][mid]==target)

        return true;

    if(matrix[i][mid]>target)

        end=mid-1;

    else

        start=mid+1;

}

}

}

return false;

}

};

```

Accepted 130 / 130 testcases passed
Nabeel Akhtar submitted at Feb 14, 2025 00:48

Runtime: 60 ms | Beats: 29.64%
Memory: 18.80 MB | Beats: 36.97%

```

1 class Solution {
2 public:
3     bool searchMatrix(vector<vector<int>>& matrix, int target) {
4         for(int i=0;i<matrix.size();i++){
5             if(matrix[i][0]<=target && matrix[i][matrix[i].size()-1]>=target){
6                 int start=0;
7                 int end=matrix[i].size()-1;
8                 while(start<=end){
9                     int mid=start+(end-start)/2;
10                    if(matrix[i][mid]==target)
11                        return true;
12                    if(matrix[i][mid]>target)
13                        end=mid-1;
14                    else
15                        start=mid+1;
16                }
17            }
18        }
19        return false;
20    }
21 };

```

Testcase: Accepted Runtime: 3 ms

Case 1 Case 2

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]]

target =
5

75. Sort Colours

```
class Solution {  
public:  
    void sortColors(vector<int>& a) {  
        int zeros=0,ones=0,twos=a.size()-1;  
        while(ones<=twos){  
            if(a[ones]==0){  
                swap(a[ones],a[zeros]);  
                zeros++;  
                ones++;  
            }  
            else if(a[ones]==2){  
                swap(a[ones],a[twos]);  
                twos--;  
            }  
            else{  
                ones++;  
            }  
        }  
    }  
}
```

};

The screenshot displays a LeetCode submission for the 'Sort Colors' problem. The submission is accepted, with a runtime of 0 ms and memory usage of 11.67 MB. The test result shows an input of [2, 0, 2, 1, 1, 0] and an output of [0, 0, 1, 1, 2, 2].

```
1 class Solution {
2 public:
3     void sortColors(vector<int>& a) {
4         int zeros=0, ones=0, twos=a.size()-1;
5         while(ones<twos){
6             if(a[ones]==0){
7                 swap(a[ones],a[zeros]);
8                 zeros++;
9                 ones++;
10            }
11            else if(a[ones]==2){
12                swap(a[ones],a[twos]);
13                twos--;
14            }
15        }
16    }
17 }
```

162. Find Peak Element

```
class Solution {
```

```
public:
```

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

```
        if(nums.size()==1)
```

```
            return 0;
```

```
        if(nums[0]>nums[1])
```

```
            return 0;
```

```
        if(nums[nums.size()-1]>nums[nums.size()-2])
```

```
            return nums.size()-1;
```

```
        int start=0;
```

```
        int end=nums.size()-1;
```

```
        while(start<=end){
```

```
            int mid=start+(end-start)/2;
```

```
            if(nums[mid]>nums[mid-1] && nums[mid]>nums[mid+1])
```

```
                return mid;
```



```

        if(start==0)

        start++;

        if(nums[mid]<nums[mid+1])

        start=mid+1;

        else

        end=mid-1;

    }

    return 0;

}

};

```

The screenshot shows a LeetCode submission for the problem "Find Peak Element". The submission is accepted, with a runtime of 0ms and memory usage of 12.53 MB. The code is in C++ and implements a binary search algorithm. The test result shows the input [1, 2, 3, 1] and the output 2.

Code:

```

1 class Solution {
2 public:
3     int findPeakElement(vector<int>& nums) {
4         if(nums.size()==1)
5             return 0;
6         if(nums[0]>nums[1])
7             return 0;
8         if(nums[nums.size()-1]>nums[nums.size()-2])
9             return nums.size()-1;
10        int start=0;
11        int end=nums.size()-1;
12        while(start<end){
13            int mid=(start+end)/2;

```

Test Result:

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums = [1, 2, 3, 1]

Output

2

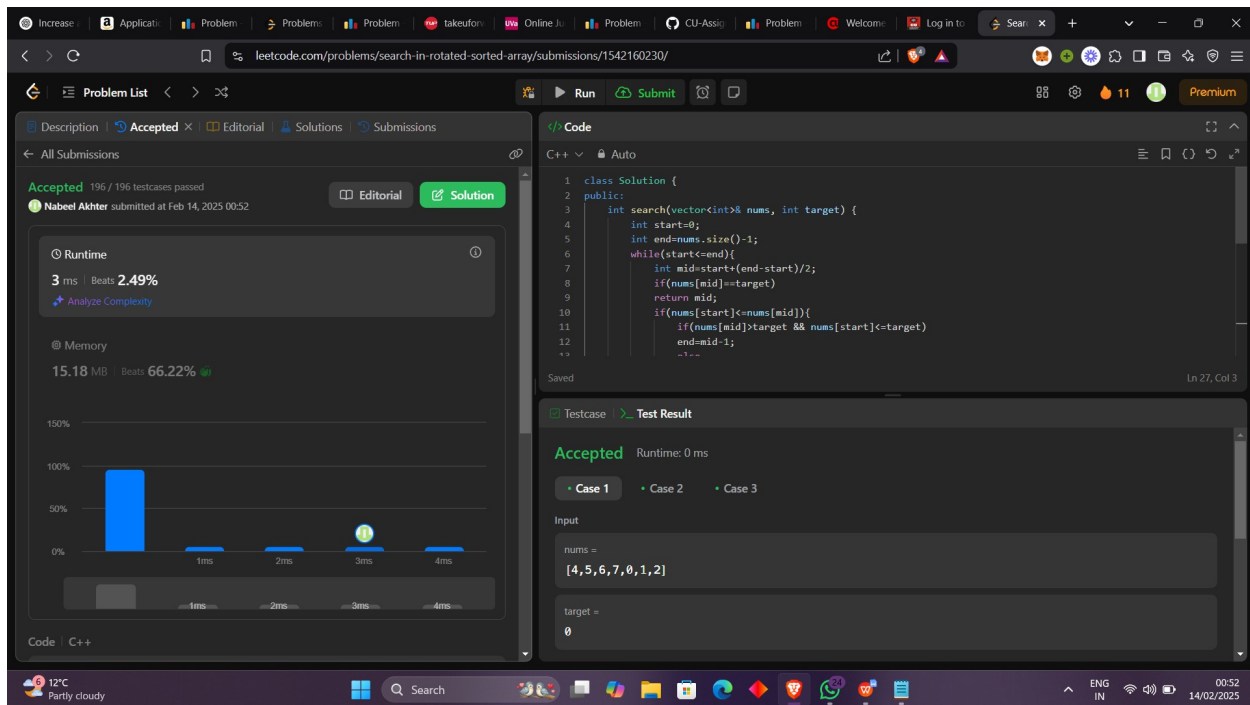
33. Search in Rotated Sorted Array

```

class Solution {
public:
    int search(vector<int>& nums, int target) {

```

```
int start=0;
int end=nums.size()-1;
while(start<=end){
    int mid=start+(end-start)/2;
    if(nums[mid]==target)
        return mid;
    if(nums[start]<=nums[mid]){
        if(nums[mid]>target && nums[start]<=target)
            end=mid-1;
        else
            start=mid+1;
    }
    else{
        cout<<mid<<endl;
        if(nums[mid]<target && nums[end]>=target)
            start=mid+1;
        else
            end=mid-1;
    }
    cout<<start<<" "<<end<<endl;
}
return -1;
}
};
```



4. Median of Two Sorted Arrays

```
class Solution {
```

```
public:
```

```
    double findMedianSortedArrays(vector<int>& nums1, vector<int>& nums2) {
```

```
        int midder=(nums1.size()+nums2.size())/2;
```

```
        int n=nums1.size();
```

```
        int m=nums2.size();
```

```
        nums1.push_back(INT_MAX);
```

```
        nums2.push_back(INT_MAX);
```

```
        if((n+m)%2==0){
```

```
            if(n==0){
```

```
                return (nums2[m/2]+nums2[m/2-1])/2.0;
```

```
            }
```

```

if(m==0){
    return (nums1[n/2]+nums1[n/2-1])/2.0;
}
int start=0;
int end=n;
int mid1,mid2;
while(start<=end){
    mid1=start+(end-start)/2;
    mid2=midder-mid1;
    if(mid2<0){
        end=mid1-1;
    }
    else if(m<mid2){
        start=mid1+1;
    }
    else{
        if(mid1==0){
            if(nums2[mid2-1]<=nums1[mid1]){
                return (nums2[mid2-1]+min(nums1[mid1],nums2[mid2]))/2.0;
            }
            else{
                start=mid1+1;
            }
        }
        else if(mid2==0){
            if(nums1[mid1-1]<=nums2[mid2]){
                return (nums1[mid1-1]+min(nums1[mid1],nums2[mid2]))/2.0;
            }

```

```

        else{
            end=mid1-1;
        }
    }
    else{
        if(nums1[mid1-1]>nums2[mid2]){
            end=mid1-1;
        }
        else if(nums2[mid2-1]>nums1[mid1]){
            start=mid1+1;
        }
        else{
            return (max(nums1[mid1-1],nums2[mid2-1])+min(nums1[mid1],nums2[mid2]))/2.0;
        }
    }
}

}

else{
    if(n==0){
        return nums2[m/2];
    }
    if(m==0){
        return nums1[n/2];
    }
    midder++;
    int start=0;
    int end=n;

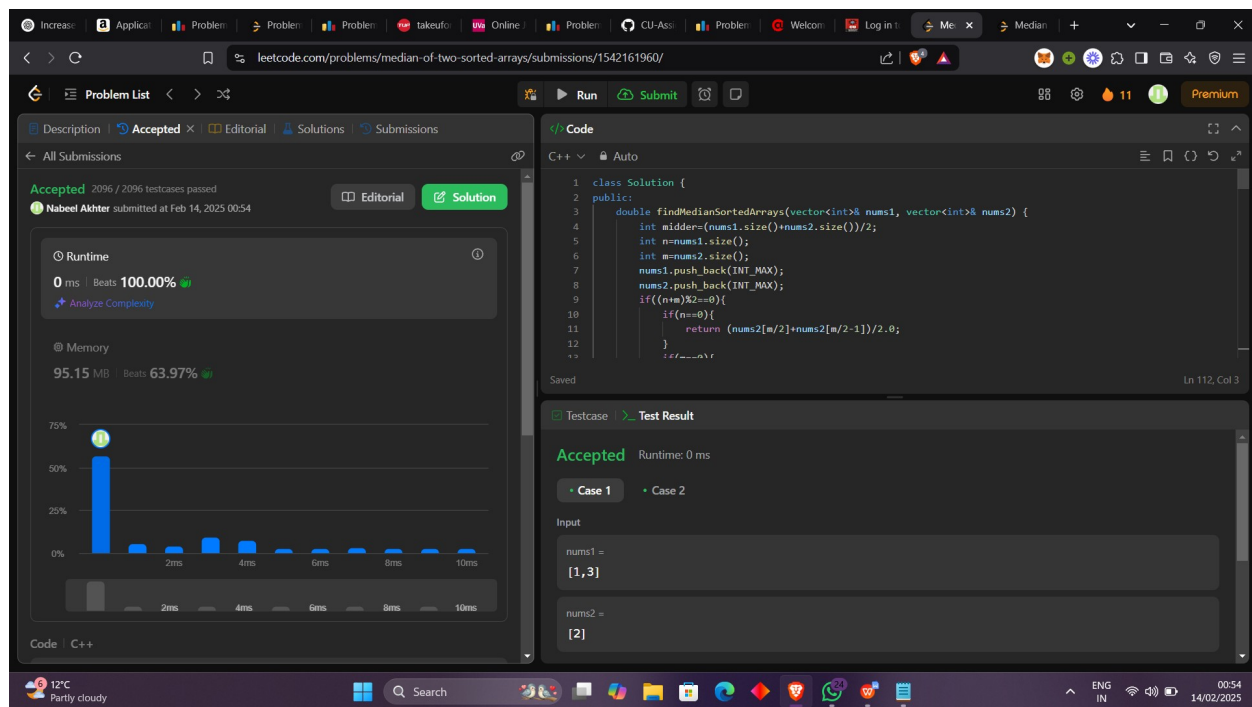
```

```

int mid1,mid2;
while(start<=end){
    mid1=start+(end-start)/2;
    mid2=midder-mid1;
    if(mid2<0){
        end=mid1-1;
    }
    else if(m<mid2){
        start=mid1+1;
    }
    else{
        if(mid1==0){
            if(nums2[mid2-1]<=nums1[mid1]){
                return nums2[mid2-1];
            }
            else{
                start=mid1+1;
            }
        }
        else if(mid2==0){
            if(nums1[mid1-1]<=nums2[mid2]){
                return nums1[mid1-1];
            }
            else{
                end=mid1-1;
            }
        }
    }
}

```

```
    else{
        if(nums1[mid1-1]>nums2[mid2]){
            end=mid1-1;
        }
        else if(nums2[mid2-1]>nums1[mid1]){
            start=mid1+1;
        }
        else{
            return max(nums2[mid2-1],nums1[mid1-1]);
        }
    }
}
}
}
return 1;
}
};
```



278. First Bad Version

```
class Solution {
```

```
public:
```

```
    int firstBadVersion(int n) {
```

```
        int first = 1;
```

```
        int last = n;
```

```
        while (first < last) {
```

```
            int mid = first + (last - first) / 2;
```

```
            if (isBadVersion(mid)) {
```

```
                last = mid;
```

```
            } else {
```

```
                first = mid + 1;
```

```
            }
```

```
        }
```

```
        return first;
```


The screenshot displays a web browser window with the LeetCode website. The main content area shows the 'firstBadVersion' problem submission page. The submission is marked as 'Accepted' with a runtime of 3ms and memory usage of 7.94 MB. The code is written in C++ and implements a binary search algorithm. The test case shows n=5 and bad=4.

Problem List < > > >

Description Accepted Editorial Solutions Submissions

All Submissions

Accepted 24 / 24 testcases passed

Nabeel Akhter submitted at Feb 14, 2025 00:55

Editorial Solution

Runtime

3 ms Beats 18.85%

Analyze Complexity

Memory

7.94 MB Beats 38.88%

60%

40%

20%

0%

1ms 2ms 3ms 4ms

Code C++

Code

```
1 class Solution {
2 public:
3     int firstBadVersion(int n) {
4         int first = 1;
5         int last = n;
6         while (first < last) {
7             int mid = first + (last - first) / 2;
8             if (isBadVersion(mid)) {
9                 last = mid;
10            } else {
11                first = mid + 1;
12            }
13        }
14    }
15 }
```

Saved Ln 16, Col 3

Testcase **Test Result**

Accepted Runtime: 0 ms

Case 1 Case 2

Input

n = 5

bad = 4

12°C Partly cloudy

Search

ENG IN 00:55 14/02/2025