**LAB-02**

**TASK-01:**

**1.QUICK SORT(pivot at end)-**

#include <iostream>

using namespace **std**;

int **partition**(int arr[],int low,int high){

    int pivot = arr[high];

    int i = low-1;

    for(int j= low; j<high ; j++){

        if(pivot>arr[j]){

            i++;

**swap**(arr[i],arr[j]);

        }

    }

**swap**(arr[i+1] , arr[high]);

    return i+1;

}

void **quicksort**(int arr[] , int low , int high){

    if(low<high){

        int pivot = **partition**(arr,low,high);

**quicksort**(arr,low,pivot-1);

**quicksort**(arr,pivot+1,high);

    }

}

void **print**(int arr[],int n){

    for(int i=0;i<n;i++){

        cout**<<**arr[i]**<<**" ";

    }

    cout**<<endl**;

}

int **main**(){

    int arr[5] = {3,1,5,6,2};

    int n = 5;

**print**(arr,n);

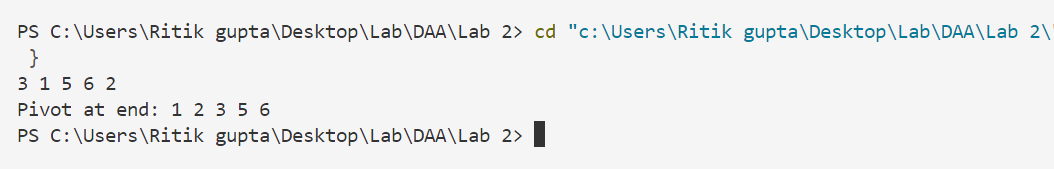
**quicksort**(arr,0,n-1);

    cout**<<**"Pivot at end: ";

**print**(arr,n);

}

Output:



**2.Quick Sort(pivot at first):-**

#include <iostream>

using namespace **std**;

int **partition**(int arr[],int low,int high){

    int pivot = arr[low];

    int j = high;

    for (int i = high; i > low; i--) {

        if (arr[i] > pivot){

**swap**(arr[i],arr[j]);

            j--;

        }

    }

**swap**(arr[j],arr[low]);

    return j;

}

void **quicksort**(int arr[] , int low , int high){

    if(low<high){

        int pivot = **partition**(arr,low,high);

**quicksort**(arr,low,pivot-1);

**quicksort**(arr,pivot+1,high);

    }

}

void **print**(int arr[],int n){

    for(int i=0;i<n;i++){

        cout**<<**arr[i]**<<**" ";

    }

    cout**<<endl**;

}

int **main**(){

    int arr[5] = {3,1,5,6,2};

    int n = 5;

**print**(arr,n);

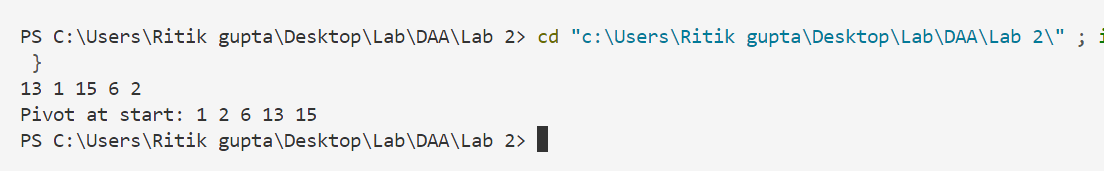
**quicksort**(arr,0,n-1);

    cout**<<**"Pivot at start: ";

**print**(arr,n);

}

Output:



3.QuickSort(pivot at middle)

#include <iostream>

using namespace **std**;

int **partition**(int arr[],int low,int high){

    int mid = (low + high)/2;

    int pivot = arr[mid];

**swap**(arr[mid] , arr[high]);

    int i = low;

    for(int j= low; j<high ; j++){

        if(pivot>arr[j]){

**swap**(arr[i],arr[j]);

            i++;

        }

    }

**swap**(arr[i] , arr[high]);

    return i;

}

void **quicksort**(int arr[] , int low , int high){

    if(low<high){

        int pivot = **partition**(arr,low,high);

**quicksort**(arr,low,pivot-1);

**quicksort**(arr,pivot+1,high);

    }

}

void **print**(int arr[],int n){

    for(int i=0;i<n;i++){

        cout**<<**arr[i]**<<**" ";

    }

    cout**<<endl**;

}

int **main**(){

    int arr[5] = {3,1,5,6,2};

    int n = 5;

**print**(arr,n);

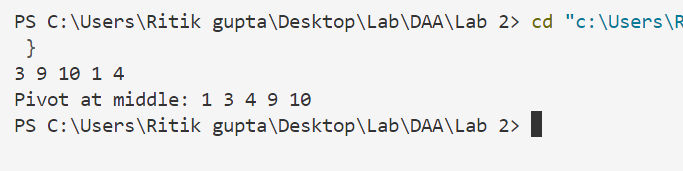
**quicksort**(arr,0,n-1);

    cout**<<**"Pivot at end: ";

**print**(arr,n);

}

**Output:**

****

**Time complexities:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Pivot** | **Start** | **Middle** | **End** |
| **Random** | **n\*logn** | **n\*logn** | **n\*logn** |
| **Best** | **n\*logn** | **n\*logn** | **n\*logn** |
| **Worst** | **n^2** | **n^2** | **n^2** |

**4.Decimal to Binary:**

#include <bits/stdc++.h>

using namespace **std**;

void **find\_bin**(int n){

    if(n==0){

        return ;

    }

**find\_bin**(n/2);

    cout**<<**n % 2;

}

int **main**(){

    int n;

    cout**<<**"Enter any number: ";

    cin**>>**n;

    int p=n;

**string** binary = "";

    if (n == 0) {

        binary **=** "0";

    } else {

        while (n > 0) {

            binary **=** char('0' + (n % 2)) **+** binary;

            n /= 2;

        }

    }

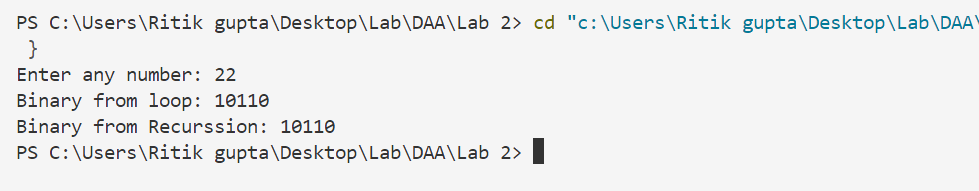
    cout**<<**"Binary from loop: "**<<**binary**<<endl**;

    cout**<<**"Binary from Recurssion: ";

**find\_bin**(p);

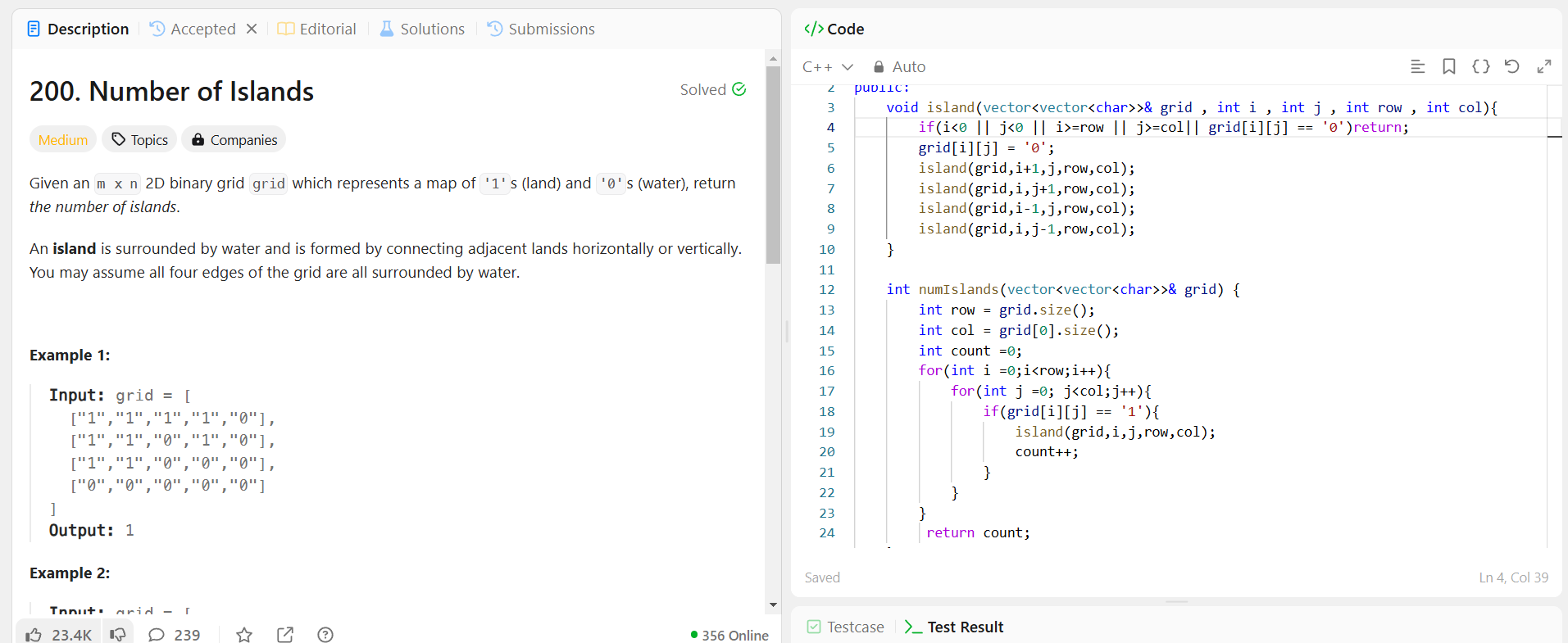
}

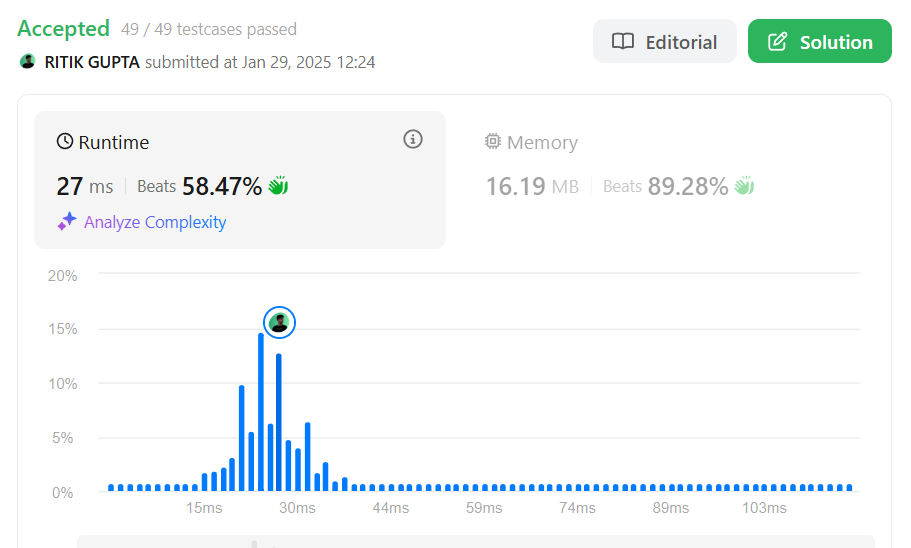
Output:



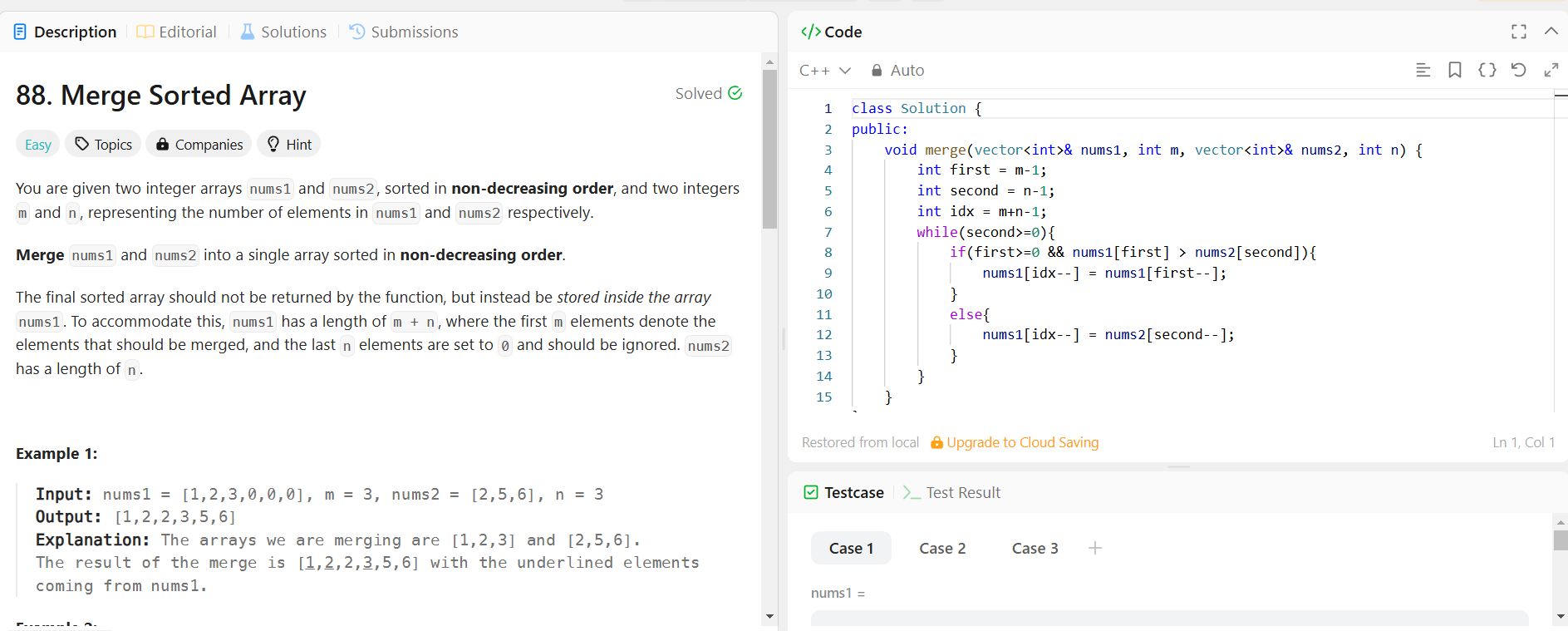
**Task-02**

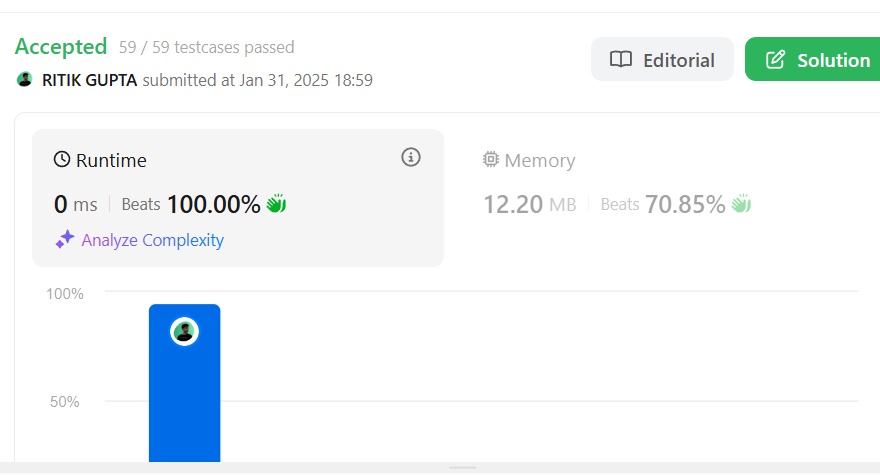
**1.Number of Islands:**

****

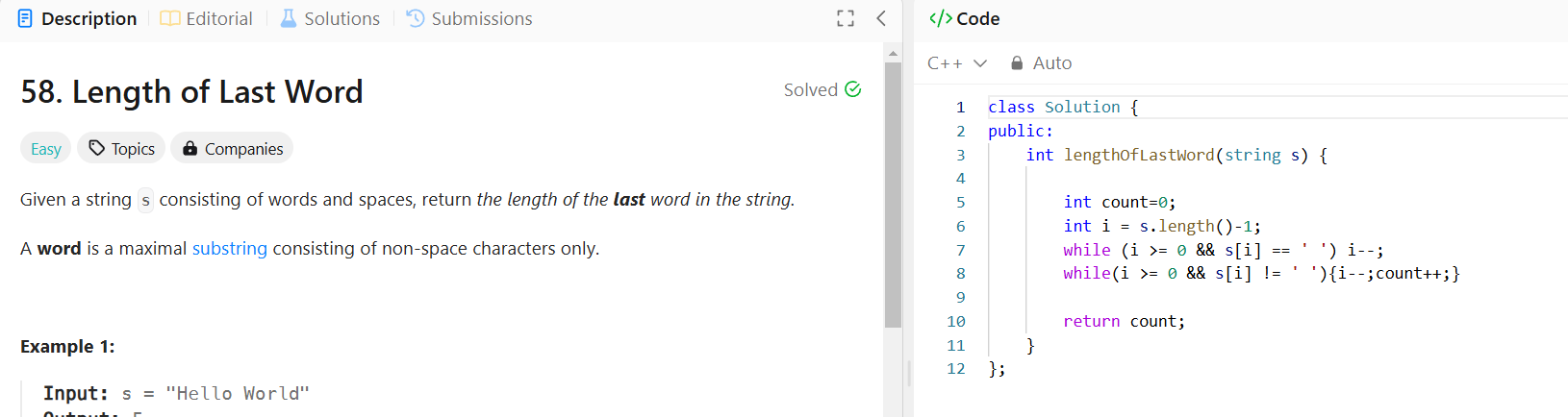
****

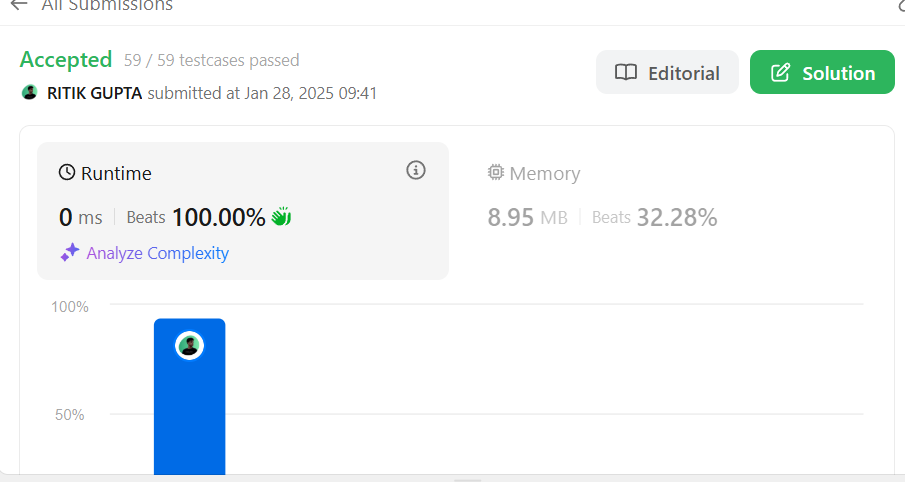
**2.Merge 2 sorted array**

****

****

**3.Length of last word**

****

****