Started on	Thursday, 29 August 2024, 11:15 AM
State	Finished
Completed on	Thursday, 3 October 2024, 10:13 AM
Time taken	34 days 22 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement

Given an array of 1s and 0s this has all 1s first followed by all 0s. Aim is to find the number of 0s. Write a program using Divide and Conquer to Count the number of zeroes in the given array.

Input Format

First Line Contains Integer m – Size of array

Next m lines Contains m numbers – Elements of an array

Output Format

First Line Contains Integer – Number of zeroes present in the given array.

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int Zeros(int arr[], int start, int end);
int Zeros(int arr[], int start, int end) {
    if(start > end){
        return 0;
    }
    int mid = (start + end) / 2;
    if (arr[mid] == 0) {
        if(arr[mid-1]!=0){
            int rc = end - mid + 1;
            return rc;
        }
        else{
            int lc = Zeros(arr, 0 , mid-1);
            int rc = end - mid + 1;
            return lc+rc;
        }
    }
    else
        return Zeros(arr, mid + 1, end);
    }
    int main() {
    int n;
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++)
        scanf("%d",&arr[i]);
    int zeroCount = Zeros(arr,0, n-1);
    printf("%d", zeroCount);
    return 0;
}
/*#include <stdio.h>
int main () {
    int m;
    scanf("%d", &m);
    int arr[m];
    for (int i=0; i<m; i++) {
        scanf("%d\n", &arr[i]);
    }
    int low=0;
    int count=0;
    int high=m-1;
    while (low<high) {
        int mid=(low+high)/2;
        if (arr[mid]==0) {
            for (int i=low; i<=high; i++) {
                if (arr[i]==0) {
                    count+=1;
                    high=mid+1;

                }
```

```
52          }
```

	Input	Expected	Got	
✔	5 1 1 1 0 0	2	2	✔
✔	10 1 1 1 1 1 1 1 1 1 1	0	0	✔
✔	8 0 0 0 0 0 0 0 0	8	8	✔
✔	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0	2	2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◀ 5-G-Product of Array elements-Minimum

Jump to...

2-Majority Eleme

Started on	Thursday, 5 September 2024, 10:20 AM
State	Finished
Completed on	Thursday, 3 October 2024, 10:15 AM
Time taken	27 days 23 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Given an array `nums` of size `n`, return *the majority element*.

The majority element is the element that appears more than `⌊n / 2⌋` times. You may assume that the majority element always exis the array.

Example 1:

```
Input: nums = [3,2,3]
Output: 3
```

Example 2:

```
Input: nums = [2,2,1,1,1,2,2]
Output: 2
```

Constraints:

- `n == nums.length`
- `1 <= n <= 5 * 10^4`
- `-2^31 <= nums[i] <= 2^31 - 1`

For example:

Input	Result
3 3 2 3	3
7 2 2 1 1 1 2 2	2

Answer: (penalty regime: 0 %)

Ace editor not ready. Perhaps reload page?
Falling back to raw text area.

```
#include <stdio.h>
int majority(int nums[], int low, int high)
{
    if (low==high)
        return nums[low];
    int mid = (low+high)/2;
    int left=majority(nums, low, mid);
    int right=majority(nums, mid + 1,high);
    if (left==right)
        return left;
    int leftCount=0;
    for (int i=low;i<=high;i++)
        if (nums[i] == left)
            leftCount++;
    int rightCount=0;
    for (int i=low;i<=high;i++)
        if (nums[i]==right)
            rightCount++;
```

	Input	Expected	Got	
✔	3 3 2 3	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◀ 1-Number of Zeros in a Given Array

Jump to...

3-Finding Floor Val

Started on	Thursday, 12 September 2024, 10:20 AM
State	Finished
Completed on	Thursday, 19 September 2024, 10:33 AM
Time taken	7 days
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement:

Given a sorted array and a value x, the floor of x is the largest element in array smaller than or equal to x. Write divide and conquer algorithm to find floor of x.

Input Format

First Line Contains Integer n – Size of array

Next n lines Contains n numbers – Elements of an array

Last Line Contains Integer x – Value for x

Output Format

First Line Contains Integer – Floor value for x

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int main () {
    int n, x, mid;
    scanf("%d", &n);
    int arr[n];
    for(int i = 0; i < n; i++){
        scanf("%d", &arr[i]);
    }
    scanf("%d", &x);
    int low = 0;
    int high = n-1;

    while (low < high) {
        mid = (low + high)/2;
        if (arr[mid] > x) {
            high = mid - 1;
        }
        else if (arr[mid] < x) {
            low = mid + 1;
        }

        if (arr[mid] == x) {
            printf("%d", arr[mid]);
            break;
        }
        else if (arr[mid] < x) {
            printf("%d", arr[mid]);
            break;
        }
        else {
            printf("%d", arr[high]);
            break;
        }

    }
}
```

	Input	Expected	Got	
✔	6 1 2 8 10 12 19 5	2	2	✔
✔	5 10 22 85 108 129 100	85	85	✔
✔	7 3 5 7 9 11 13 15 10	9	9	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◀ 2-Majority Element

Jump to...

4-Two Elements sum t

Started on	Thursday, 12 September 2024, 11:23 AM
State	Finished
Completed on	Thursday, 19 September 2024, 11:28 AM
Time taken	7 days
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement:

Given a sorted array of integers say arr[] and a number x. Write a recursive program using divide and conquer strategy to check if exist two elements in the array whose sum = x. If there exist such two elements then return the numbers, otherwise print as “No”.

Note: Write a Divide and Conquer Solution

Input Format

First Line Contains Integer n – Size of array

Next n lines Contains n numbers – Elements of an array

Last Line Contains Integer x – Sum Value

Output Format

First Line Contains Integer – Element1

Second Line Contains Integer – Element2 (Element 1 and Elements 2 together sums to value “x”)

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int main ()
{
    int n;
    scanf ("%d", &n);
    int a[n];
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
    int x;
    scanf ("%d", &x);
    int left = 0;
    int right = n-1;
    while (left < right)
    {
        int mid = (left + right) / 2;
        for (int i = 0; i < n; i++)
        {
            if (a[i] < a[mid])
            {
                right = mid - 1;
            }
            else if (a[i] > a[mid])
            {
                left = mid + 1;
            }
        }
    }
    int flag = 0;
    for(int i = 0; i < n; i++)
    {
        for (int j = 1; j < i; j++)
        {
            if (a[i] + a[j] == x)
            {
                printf("%d\n", a[j]);
                printf("%d", a[i]);
                flag = 1;
            }
        }
    }
    if (flag == 0)
    {
        printf("No");
    }
}
```

	Input	Expected	Got	
✔	4 2 4 8 10 14	4 10	4 10	✔
✔	5 2 4 6 8 10 100	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◀ 3-Finding Floor Value

Jump to...

5-Implementation of Quick S

Started on	Thursday, 3 October 2024, 10:11 AM
State	Finished
Completed on	Thursday, 3 October 2024, 10:42 AM
Time taken	30 mins 46 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Write a Program to Implement the Quick Sort Algorithm

Input Format:
The first line contains the no of elements in the list-n
The next n lines contain the elements.

Output:
Sorted list of elements

For example:

Input	Result
5 67 34 12 98 78	12 34 67 78 98

Answer:

```c
#include <stdio.h>
void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {
        int pivot = arr[high];
        int i = (low - 1);

        for (int j = low; j < high; j++)
        {

            if (arr[j] <= pivot)
            {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        int temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;

        int p= i + 1;
        quickSort(arr, low, p- 1);
        quickSort(arr, p+ 1, high);
    }
}

int main()
{
    int n;
    scanf("%d", &n);

    int arr[n];

    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    quickSort(arr, 0, n - 1);

    for (int i = 0; i < n; i++)
    {
        printf("%d ", arr[i]);
    }
```

```
47      printf("\n");
48  }
49
```

	Input	Expected	Got	
✔	5 67 34 12 98 78	12 34 67 78 98	12 34 67 78 98	✔
✔	10 1 56 78 90 32 56 11 10 90 114	1 10 11 32 56 56 78 90 90 114	1 10 11 32 56 56 78 90 90 114	✔
✔	12 9 8 7 6 5 4 3 2 1 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 4-Two Elements sum to x

Jump to...

1-DP-Playing with Numbe