CS23331-DAA-2024-CSE / 1-Number of Zeros in a Given Array

1-Number of Zeros in a Given Array

Started on	Friday, 19 September 2025, 2:07 PM
State	Finished
Completed on	Saturday, 20 September 2025, 8:36 PM
Time taken	1 day 6 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

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 dac(int arr[], int low, int high, int n)
{
    if(high<low){
        return 0;
    }
    int mid=(low+high)/2;
    if(arr[mid]==0 && (mid==0|| arr[mid-1]==1)){
        return n-mid;
    }
    else if (arr[mid]==0){
        return dac(arr,low,mid-1,n);
    }
    else{
        return dac(arr,mid+1,high,n);
    }
}

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

    int ans=dac(arr,0,n-1,n);
    printf("%d",ans);
}
```

	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	8	8	✔

	0			
	0			
	0			
	0			
	0			
	0			
	0			
	0			
✔	17	2	2	✔
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	1			
	0			
	0			

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Back to Course

CS23331-DAA-2024-CSE / 2-Majority Element

2-Majority Element

Started on	Friday, 19 September 2025, 2:22 PM
State	Finished
Completed on	Saturday, 20 September 2025, 8:50 PM
Time taken	1 day 6 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

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

The majority element is the element that appears more than $\lfloor n / 2 \rfloor$ times. You may assume that the majority element always exists in 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 %)

```c
1   // #include <stdio.h>
2
3   // int majorityElement(int arr[], int n) {
4   //     int count = 0, candidate = 0;
5
6   //     for (int i = 0; i < n; i++) {
7   //         if (count == 0) {
8   //             candidate = arr[i];
9   //         }
10  //         if (arr[i] == candidate)
11  //             count++;
12  //         else
13  //             count--;
14  //     }
15
16  //     return candidate;
17  // }
18
19  // int main() {
20  //     int n;
21  //     scanf("%d", &n);
22
23  //     int arr[n];
24  //     for (int i = 0; i < n; i++) {
25  //         scanf("%d", &arr[i]);
26  //     }
27
28  //     printf("%d\n", majorityElement(arr, n));
29  // }
30
31  #include <stdio.h>
32  #include <stdlib.h>
33
34  int cmpfunc(const void* a, const void* b) {
35      return (*(int*)a - *(int*)b);
36  }
37
38  int majorityElement(int* nums, int n) {
39      qsort(nums, n, sizeof(int), cmpfunc);
40      return nums[n / 2];
41  }
```

```
42
43    int main() {
44        int n;
45        scanf("%d", &n);
46
47        int nums[n];
48        for (int i = 0; i < n; i++) {
49            scanf("%d", &nums[i]);
50        }
51
52        printf("%d\n", majorityElement(nums, n));
```

	Input	Expected	Got	
✔	3 3 2 3	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Finish review

Back to Course

CS23331-DAA-2024-CSE / 3-Finding Floor Value

3-Finding Floor Value

Started on	Saturday, 20 September 2025, 8:51 PM
State	Finished
Completed on	Saturday, 20 September 2025, 9:04 PM
Time taken	13 mins 22 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 | Correct   Mark 1.00 out of 1.00 | ⚑ Flag question

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

Chat with th

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int findFloor(int arr[], int n, int x) {
    int low = 0, high = n - 1;
    int floorVal = 0;

    while (low <= high) {
        int mid = (low + high) / 2;

        if (arr[mid] == x) {
            return arr[mid];
        }
        else if (arr[mid] < x) {
            floorVal = arr[mid];
            low = mid + 1;
        }
        else {
            high = mid - 1;
        }
    }

    return floorVal;
}

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

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

    scanf("%d", &x);

    int floorVal = findFloor(arr, n, x);
    printf("%d\n", floorVal);

    return 0;
}
```

	Input	Expected	Got	
✔	6 1 2 8 10 12 19 5	2	2	✔
✔	5 10 22 	85	85	✔

	85		
	108		
	129		
	100		
✔	7	9	9 ✔
	3		
	5		
	7		
	9		
	11		
	13		
	15		
	10		

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Finish review

Back to Course

CS23331-DAA-2024-CSE / 4-Two Elements sum to x

4-Two Elements sum to x

Started on	Saturday, 20 September 2025, 9:04 PM
State	Finished
Completed on	Saturday, 20 September 2025, 9:21 PM
Time taken	16 mins 56 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

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 there 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 dac(int arr[],int low, int high, int x, int *a, int *b){
    if(high<low)
        return 0;

    int sum=arr[low]+arr[high];

    if(sum==x){
        *a=arr[low];
        *b=arr[high];
        return 1;
    }
    else if(sum<x){
        return dac(arr,low+1,high,x,a,b);
    }
    else{
        return dac(arr,low,high-1,x,a,b);
    }
}

int main()
{
    int n;
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    int x,a,b;
    scanf("%d",&x);
    if(dac(arr,0,n-1,x,&a,&b)){
        printf("%d\n%d",a,b);
    }
    else{
        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.

Back to Course

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Back to Course

CS23331-DAA-2024-CSE / 5-Implementation of Quick Sort

5-Implementation of Quick Sort

Started on	Saturday, 20 September 2025, 9:21 PM
State	Finished
Completed on	Saturday, 20 September 2025, 9:24 PM
Time taken	3 mins 2 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

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
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   int cmpfunc(const void *a, const void *b) {
5       return (*(int*)a - *(int*)b);
6   }
7
8   int main() {
9       int n;
10      scanf("%d", &n);
11
12      int arr[n];
13      for (int i = 0; i < n; i++) {
14          scanf("%d", &arr[i]);
15      }
16
17      qsort(arr, n, sizeof(int), cmpfunc);
18
19      for (int i = 0; i < n; i++) {
20          printf("%d ", arr[i]);
21      }
22  }
23
```

	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.

Finish review

Back to Course