

1-Number of Zeros in a Given Array

Started on Wednesday, 17 September 2025, 3:27 PM

State Finished

Completed on Wednesday, 17 September 2025, 3:40 PM

Time taken 12 mins 37 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 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.



Search



ENG
IN



09:06 PM

11-10-2025

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int countZeroes(int arr[], int low, int high) {
4     if (low > high) return 0;
5     if (arr[high] == 1) return 0;
6     if (arr[low] == 0) return (high - low + 1);
7     int mid = (low + high) / 2;
8     int leftCount = countZeroes(arr, low, mid);
9     int rightCount = countZeroes(arr, mid + 1, high);
10    return leftCount + rightCount;
11 }
12
13 int main() {
14     int m;
15     scanf("%d", &m);
16     int arr[m];
17     for (int i = 0; i < m; i++) {
18         scanf("%d", &arr[i]);
19     }
20     int result = countZeroes(arr, 0, m - 1);
21     printf("%d\n", result);
22     return 0;
23 }
```




2-Majority Element

Started on	Wednesday, 17 September 2025, 3:40 PM
State	Finished
Completed on	Wednesday, 17 September 2025, 3:45 PM
Time taken	4 mins 38 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

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

The majority element is the element that appears more than $\lfloor \frac{n}{2} \rfloor$ times. You may assume that the majority element always exists in the array.

Example 1:

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

Output: 3

Example 2:

Example 1:

Input: nums = [3,2,3]

Output: 3

Example 2:

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

Output: 2

Constraints:

- $n == \text{nums.length}$
- $1 \leq n \leq 5 * 10^4$
- $-2^{31} \leq \text{nums}[i] \leq 2^{31} - 1$

For example:

Input	Result
3	3
3 2 3	

Input	Result
7	2
2 2 1 1 1 2 2	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int majorityElement(int* nums, int n) {
4     int count = 0, candidate = 0;
5     for (int i = 0; i < n; i++) {
6         if (count == 0) {
7             candidate = nums[i];
8             count = 1;
9         } else if (nums[i] == candidate) {
10            count++;
11        } else {
12            count--;
13        }
14    }
15    return candidate;
16 }
17
18 int main() {
19     int n;
20     scanf("%d", &n);
21     int arr[n];
22     for (int i = 0; i < n; i++) {
23         scanf("%d", &arr[i]);
24     }
25     int res = majorityElement(arr, n);
26     printf("%d\n", res);
27     return 0;
28 }
```

	Input	Expected	Got	
✓	3 3 2 3	3	3	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



3-Finding Floor Value

Started on Wednesday, 17 September 2025, 3:46 PM

State Finished

Completed on Wednesday, 17 September 2025, 3:46 PM

Time taken 37 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

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int findFloor(int arr[], int low, int high, int x) {
4     if (low > high) return -1;
5     int mid = (low + high) / 2;
6
7     if (arr[mid] == x) return arr[mid];
8     else if (arr[mid] > x) return findFloor(arr, low, mid - 1, x);
9     else {
10         int floorRight = findFloor(arr, mid + 1, high, x);
11         if (floorRight == -1) return arr[mid];
12         else return floorRight;
13     }
14 }
15
16 int main() {
17     int n;
18     scanf("%d", &n);
19     int arr[n];
20     for(int i=0; i<n; i++) {
21         scanf("%d", &arr[i]);
22     }
23     int x;
24     scanf("%d", &x);
25     int res = findFloor(arr, 0, n-1, x);
26     printf("%d\n", res);
27     return 0;
28 }
29 }
```

	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	✓

4-Two Elements sum to x

Started on	Wednesday, 17 September 2025, 3:47 PM
State	Finished
Completed on	Wednesday, 17 September 2025, 3:48 PM
Time taken	57 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

First Line Contains Integer – Element1

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

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int findPairSum(int arr[], int left, int right, int x) {
4     if (left >= right) return 0;
5     int sum = arr[left] + arr[right];
6     if (sum == x) {
7         printf("%d\n%d\n", arr[left], arr[right]);
8         return 1;
9     } else if (sum < x) {
10        return findPairSum(arr, left + 1, right, x);
11    } else {
12        return findPairSum(arr, left, right - 1, x);
13    }
14 }
15
16 int main() {
17     int n;
18     scanf("%d", &n);
19     int arr[n];
20     for(int i=0; i<n; i++) {
21         scanf("%d", &arr[i]);
22     }
23     int x;
24     scanf("%d", &x);
25     if (!findPairSum(arr, 0, n-1, x)) {
26         printf("No\n");
27     }
28     return 0;
29 }
30
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

5-Implementation of Quick Sort

Started on	Wednesday, 17 September 2025, 3:48 PM
State	Finished
Completed on	Wednesday, 17 September 2025, 3:49 PM
Time taken	39 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

Output:

Sorted list of elements

For example:

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

Answer:

```
1 #include <stdio.h>
2
3 void swap(int *a, int *b) {
4     int temp = *a;
5     *a = *b;
6     *b = temp;
7 }
8
9 int partition(int arr[], int low, int high) {
10    int pivot = arr[high];
11    int i = low - 1;
12    for(int j = low; j < high; j++) {
13        if(arr[j] <= pivot) {
14            i++;
15            swap(&arr[i], &arr[j]);
16        }
17    }
18    swap(&arr[i + 1], &arr[high]);
19    return i + 1;
20 }
21 }
```

```
20 }
21
22 void quickSort(int arr[], int low, int high) {
23     if(low < high) {
24         int pi = partition(arr, low, high);
25         quickSort(arr, low, pi - 1);
26         quickSort(arr, pi + 1, high);
27     }
28 }
29
30 int main() {
31     int n;
32     scanf("%d", &n);
33     int arr[n];
34     for(int i = 0; i < n; i++) {
35         scanf("%d", &arr[i]);
36     }
37     quickSort(arr, 0, n - 1);
38     for(int i = 0; i < n; i++) {
39         printf("%d ", arr[i]);
40     }
41     printf("\n");
42     return 0;
43 }
44 }
```

	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.