

1-Number of Zeros in a Given Array

Started on Tuesday, 9 September 2025, 12:11 PM

State Finished

Completed on Tuesday, 9 September 2025, 12:14 PM

Time taken 2 mins 44 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.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3+ int countZeroes(int arr[], int low, int high) {
4+     if (low > high) {
5+         return 0;
6+     }
7+     int mid = low + (high - low) / 2;
8+     if (arr[mid] == 0) {
9+         return (high - mid + 1) + countZeroes(arr, low, mid - 1);
10+    } else {
11+        return countZeroes(arr, mid + 1, high);
12+    }
13 }
```

```

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

	Input	Expected	Got	
✓	5 1 1 1 0 0	2	2	✓
✓	10 1 1 1 1 1 1 1 1 1	0	0	✓
✓	8	8	8	✓

	*				
	1				
	1				
	1				
	1				
✓	8	8	8	✓	
	0	0	0		
	0	0	0		
	0	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		
	1	1	1		
	1	1	1		
	0	0	0		
	0	0	0		

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Finish review](#)

2-Majority Element

Started on	Tuesday, 9 September 2025, 12:14 PM
State	Finished
Completed on	Tuesday, 9 September 2025, 12:19 PM
Time taken	5 mins 33 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 $\lfloor n / 2 \rfloor$ 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 * 104`
- $-2^{31} \leq \text{nums}[i] \leq 2^{31} - 1$

For example:

Input	Result
-------	--------

3 2 3	
7	2
2 2 1 1 1 2 2	

Answer: (penalty regime: 0 %)

```

1 #include <stdio.h>
2
3 int majorityElement(int* nums, int numsSize) {
4     int majorityCount = numsSize / 2;
5     for (int i = 0; i < numsSize; i++) {
6         int count = 0;
7         for (int j = 0; j < numsSize; j++) {
8             if (nums[j] == nums[i]) {
9                 count++;
10            }
11        }
12        if (count > majorityCount) {
13            return nums[i];
14        }
15    }
16    return -1;
17 }
18
19 int main() {
20     int n;
21     scanf("%d", &n);
22
23     int nums[n];
24     for (int i = 0; i < n; i++) {
25         scanf("%d", &nums[i]);
26     }
27     int result = majorityElement(nums, n);
28     printf("%d\n", result);
29     return 0;
30 }
```

	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	Tuesday, 9 September 2025, 12:20 PM
State	Finished
Completed on	Tuesday, 9 September 2025, 12:21 PM
Time taken	1 min 45 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.     int floor = -1;
5.     while (low <= high) {
6.         int mid = low + (high - low) / 2;
7.         if (arr[mid] == x) {
8.             return arr[mid];
9.         } else if (arr[mid] < x) {
10.             floor = arr[mid];
11.             low = mid + 1;
12.         } else {
13.             high = mid - 1;
14.         }
15.     }
16. }
```

```

2
3 int findFloor(int arr[], int low, int high, int x) {
4     int floor = -1;
5     while (low <= high) {
6         int mid = low + (high - low) / 2;
7         if (arr[mid] == x) {
8             return arr[mid];
9         } else if (arr[mid] < x) {
10            floor = arr[mid];
11            low = mid + 1;
12        } else {
13            high = mid - 1;
14        }
15    }
16    return floor;
17 }
18
19 int main() {
20     int n, x;
21     scanf("%d", &n);
22     int arr[n];
23     for (int i = 0; i < n; i++) {
24         scanf("%d", &arr[i]);
25     }
26     scanf("%d", &x);
27     int floorValue = findFloor(arr, 0, n - 1, x);
28     printf("%d\n", floorValue);
29     return 0;
30 }
```

	Input	Expected	Got	
✓	6 1 2 8 10 12 19 5	2	2	✓
✓	5 10 22 85 108 129 100	85	85	✓

	Input	Expected	Got	
✓	6 1 2 8 10 12 19 5		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.

[Finish review](#)

4-Two Elements sum to x

Started on	Tuesday, 9 September 2025, 12:22 PM
State	Finished
Completed on	Tuesday, 9 September 2025, 12:24 PM
Time taken	1 min 44 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 %)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     int n, x;
6     scanf("%d", &n);
7     int *arr = malloc(n * sizeof(int));
8     for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
9     scanf("%d", &x);
10
11     int i = 0, j = n - 1;
```

```

4. int main() {
5     int n, x;
6     scanf("%d", &n);
7     int *arr = malloc(n * sizeof(int));
8     for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
9     scanf("%d", &x);
10
11    int i = 0, j = n - 1;
12    while (i < j) {
13        int sum = arr[i] + arr[j];
14        if (sum == x) {
15            printf("%d\n%d\n", arr[i], arr[j]);
16            free(arr);
17            return 0;
18        } else if (sum < x) {
19            i++;
20        } else {
21            j--;
22        }
23    }
24
25    printf("No\n");
26    free(arr);
27    return 0;
28 }

```

	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.

5-Implementation of Quick Sort

Started on Tuesday, 9 September 2025, 12:24 PM

State Finished

Completed on Tuesday, 9 September 2025, 12:25 PM

Time taken 45 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	12 34 67 78 98
67 34 12 98 78	

Answer:

```
1 #include <stdio.h>
2
3 void swap(int *a, int *b) {
4     int t = *a; *a = *b; *b = t;
5 }
6
7 int partition(int arr[], int low, int high) {
8     int pivot = arr[high], i = low - 1;
9     for (int j = low; j < high; j++) {
10         if (arr[j] < pivot) swap(&arr[j], &arr[i]);
11     }
12 }
```

```

1 // quicksort.c
2
3 void swap(int *a, int *b) {
4     int t = *a; *a = *b; *b = t;
5 }
6
7 int partition(int arr[], int low, int high) {
8     int pivot = arr[high], i = low - 1;
9     for (int j = low; j < high; j++) {
10         if (arr[j] < pivot) swap(&arr[++i], &arr[j]);
11     swap(&arr[i + 1], &arr[high]);
12     return i + 1;
13 }
14
15 void quickSort(int arr[], int low, int high) {
16     if (low < high) {
17         int pi = partition(arr, low, high);
18         quickSort(arr, low, pi - 1);
19         quickSort(arr, pi + 1, high);
20     }
21 }
22
23 int main() {
24     int n;
25     scanf("%d", &n);
26     int arr[n];
27     for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
28     quickSort(arr, 0, n - 1);
29     for (int i = 0; i < n; i++) printf("%d ", arr[i]);
30     return 0;
31 }

```

	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](#)