



**Started on** Wednesday, 17 September 2025, 8:25 AM

**State** Finished

**Completed on** Wednesday, 17 September 2025, 8:28 AM

**Time taken** 3 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**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  int main(){
3      int m,c=0;
4      scanf("%d",&m);
5      int a[m];
6      for(int i=0;i<m;i++){
7          scanf("%d",&a[i]);
8          if(a[i]==0)
9              c++;
10     }
11     printf ( " % d " , c ) ;
12 }
```

	Input	Expected	Got	
↑	5 1 1 1 0 0	2	2	↑
↑	10 1 1 1 1 1 1 1 1 1 1 1	0	0	↑

	Input	Expected	Got	
↑	8 0 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 1 1 1 0 0	2	2	↑

Passed all tests! ↑

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



**Started on** Wednesday, 17 September 2025, 8:29 AM

**State** Finished

**Completed on** Wednesday, 17 September 2025, 8:47 AM

**Time taken** 17 mins 36 secs

**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 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`
- `-231 <= nums[i] <= 231 - 1`

For example:


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

Answer: (penalty regime: 0 %)

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

	Input	Expected	Got	
↑	3 3 2 3	3	3	↑



Passed all tests! 

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



**Started on** Wednesday, 17 September 2025, 8:47 AM

**State** Finished

**Completed on** Wednesday, 17 September 2025, 8:56 AM

**Time taken** 9 mins 17 secs

**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 %)

```

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

```

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



	Input	Expected	Got	
↑	7	9	9	↑
	3			
	5			
	7			
	9			
	11			
	13			
	15			
	10			

Passed all tests! ↑

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



**Started on** Wednesday, 17 September 2025, 8:57 AM

**State** Finished

**Completed on** Wednesday, 17 September 2025, 9:01 AM

**Time taken** 4 mins 36 secs

**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 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  int findPair(int arr[], int left, int right, int x, int* a, int* b) {
3      if (left >= right)
4          return 0;
5      int sum = arr[left] + arr[right];
6      if (sum == x) {
7          *a = arr[left];
8          *b = arr[right];
9          return 1;
10     } else if (sum < x) {
11         return findPair(arr, left + 1, right, x, a, b);
12     } else {
13         return findPair(arr, left, right - 1, x, a, b);
14     }
15 }
16 int
17 main(int m, x;
18 ) { scanf("%d", &n);
19     int arr[n];
20     for(int i = 0; i < n; i++){
21         scanf("%d", &arr[i]);
22     }
23     scanf("%d", &x);
24     int a, b;
25     if (findPair(arr, 0, n - 1, x, &a, &b)) {
26         printf("%d\n%d\n", a, b);
27     } else {
28         printf("No\n");
29     }
30 }
31

```

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

Passed all tests! 

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



7



**Started on** Wednesday, 17 September 2025, 9:02 AM

**State** Finished

**Completed on** Wednesday, 17 September 2025, 9:05 AM

**Time taken** 3 mins 3 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	12 34 67 78 98
67 34 12 98 78	

Answer:

```

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

```

	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	↑

	Input	Expected	Got	
↑	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.

[Back to Course](#)