

7 RANJANI S 2024-AIDS ▾R2**Started on** Saturday, 20 September 2025, 3:31 PM**State** Finished**Completed on** Friday, 24 October 2025, 9:52 PM**Time taken** 34 days 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**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
4  int findFirstZero(int arr[], int low, int high) {
5      if (high >= low) {
6          int mid = (low + high) / 2;
7
8          if ((mid == 0 || arr[mid - 1] == 1) && arr[mid] == 0)
9              return mid;
10
11         if (arr[mid] == 1)
12             return findFirstZero(arr, mid + 1, high);
13
14         else
15             return findFirstZero(arr, low, mid - 1);
16     }
17     return -1;
18 }
19
20
21 int main() {
22     int m;
23     scanf("%d", &m);
24
25     int arr[m];
26     for (int i = 0; i < m; i++)
27         scanf("%d", &arr[i]);
28
29     int firstZeroIndex = findFirstZero(arr, 0, m - 1);
30     int count = 0;
31
32     if (firstZeroIndex == -1)
33         count = 0;
34     else
35         count = m - firstZeroIndex;
36
37     printf("%d\n", count);
38     return 0;
39 }
40
```

	Input	Expected	Got	
✓	5 1 1 1 0 0	2	2	✓
✓	10 1 1 1 1 1 1 1 1 1 1 1 1	0	0	✓
✓	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	Saturday, 20 September 2025, 3:36 PM
State	Finished
Completed on	Saturday, 25 October 2025, 4:09 PM
Time taken	35 days
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  $\lfloor n / 2 \rfloor$  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`
- `-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
3  int majorityElement(int* nums, int numsSize) {
4      int count = 0;
5      int candidate = 0;
6
7      for (int i = 0; i < numsSize; i++) {
8          if (count == 0) {
9              candidate = nums[i];
10         }
11         if (nums[i] == candidate)
12             count++;
13         else
14             count--;
15     }
16     return candidate;
17 }
18
19
20 int main() {
21     int n;
22     scanf("%d", &n);
23     int nums[n];
24
25     for (int i = 0; i < n; i++) {
26         scanf("%d", &nums[i]);
27     }

```

```
27     }
28
29     int result = majorityElement(nums, n);
30     printf("%d\n", result);
31
32     return 0;
33 }
34
```

	Input	Expected	Got	
✓	3	3	3	✓
	3 2 3			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)





RANJANI S 2024-AIDS ▾

**R2****Started on** Saturday, 11 October 2025, 3:30 PM**State** Finished**Completed on** Saturday, 11 October 2025, 3:36 PM**Time taken** 5 mins 25 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
3  int findFloor(int arr[], int low, int high, int x) {
4      if (low > high)
5          return -1;
6
7
8      if (x >= arr[high])
9          return arr[high];
10
11     int mid = (low + high) / 2;
12
13     if (arr[mid] == x)
14         return arr[mid];
15
16
17     if (mid > 0 && arr[mid - 1] <= x && x < arr[mid])
18         return arr[mid - 1];
19
20
21     if (x < arr[mid])
22         return findFloor(arr, low, mid - 1, x);
23
24
25     return findFloor(arr, mid + 1, high, x);
26 }
27
28 int main() {
29     int n, x;
30     scanf("%d", &n);
31
32     int arr[n];
33
34     for (int i = 0; i < n; i++)
35         scanf("%d", &arr[i]);
36
37
38     scanf("%d", &x);
39
40     int result = findFloor(arr, 0, n - 1, x);
41
42     if (result == -1)
43         printf("%d\n", x);
44     else
45         printf("%d\n", result);
46
47     return 0;
48 }
49

```

	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.

[Back to Course](#)





Started on	Saturday, 11 October 2025, 3:36 PM
State	Finished
Completed on	Saturday, 11 October 2025, 3:39 PM
Time taken	2 mins 27 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
3  int findPair(int arr[], int low, int high, int x, int *a, int *b) {
4      if (low >= high)
5          return 0;
6
7      int sum = arr[low] + arr[high];
8
9      if (sum == x) {
10         *a = arr[low];
11         *b = arr[high];
12         return 1;
13     }
14     else if (sum > x)
15         return findPair(arr, low, high - 1, x, a, b);
16     else
17         return findPair(arr, low + 1, high, x, a, b);
18 }
19
20 int main() {
21     int n, x;
22     scanf("%d", &n);
23
24     int arr[n];
25     for (int i = 0; i < n; i++)
26         scanf("%d", &arr[i]);
27
28     scanf("%d", &x);
29
30     int a, b;
31     if (findPair(arr, 0, n - 1, x, &a, &b)) {
32         printf("%d\n", a);
33         printf("%d\n", b);
34     } else {
35         printf("No\n");
36     }
37
38     return 0;
39 }
40

```

	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.

[Back to Course](#)

7 RANJANI S 2024-AIDS ▾R2

---

**Started on** Saturday, 11 October 2025, 3:39 PM

---

**State** Finished

---

**Completed on** Saturday, 11 October 2025, 3:42 PM

---

**Time taken** 3 mins 7 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:**

```

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
13     for (int j = low; j < high; j++) {
14         if (arr[j] <= pivot) {
15             i++;
16             swap(&arr[i], &arr[j]);
17         }
18     }
19
20     swap(&arr[i + 1], &arr[high]);
21     return (i + 1);
22 }
23
24 void quickSort(int arr[], int low, int high) {
25     if (low < high) {
26         int pi = partition(arr, low, high);
27
28         quickSort(arr, low, pi - 1);
29         quickSort(arr, pi + 1, high);
30     }
31 }
32
33 int main() {
34     int n;
35     scanf("%d", &n);
36
37     int arr[n];
38     for (int i = 0; i < n; i++)
39         scanf("%d", &arr[i]);
40
41     quickSort(arr, 0, n - 1);
42
43     for (int i = 0; i < n; i++)
44         printf("%d ", arr[i]);
45
46     return 0;

```

```
47 | }
48 |
```

	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.

[Back to Course](#)