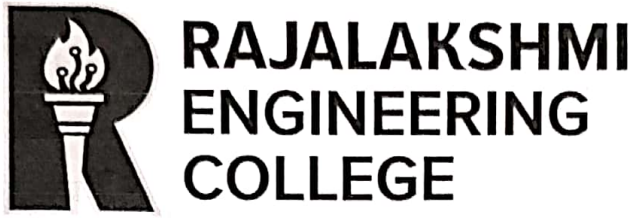
RAJALAKSHMI ENGINEERING COLLEGE
(Autonomous)

RAJALAKSHMI NAGAR, THANDALAM, CHENNAI-602105



CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS

LABORATORY RECORD NOTEBOOK

Register Number	:	241801197
Name of the Student	:	C. Parvendhan
Year / Semester	:	**II / III**
Branch	:	**Artificial Intelligence and Data Science**
Academic Year	:	**2025 – 2026 (ODD)**

RAJALAKSHMI ENGINEERING COLLEGE

[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name: C. PARVENDHAN

Academic Year: 2025-2026 (ODD) Semester: III Branch: AIDS

Register Number: 2116241801197

Certified that this is the bonafide record of work done by the above student in the **CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY** during the year 2025 - 2026.

18/11/2025

Signature of the Faculty In- charge

Submitted for the Practical Examination held on .....

Internal Examiner

RAJALAKSHMI ENGINEERING COLLEGE

(An Autonomous Institution affiliated to Anna University)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Subject: CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS

Reg No: 241801197

Name: PARVENDHAN C

Department: AIDS

INDEX

S.NO	Date	Name of the Experiment	GitHub QR Code	Marks Awarded	Signature
1.	26/7/2025	Basic C Programming		9	
2.	21/08/25	Finding Time Complexity of Algorithms		8	
3.	19/9/25	Greedy Algorithms		8	
4.	26/9/25	Divide and Conquer		9	

S.NO	Date	Name of the Experiment	GitHub QR Code	Marks Awarded	Signature
5.	24/10/25	Dynamic Programming		8	
6.	31/10/25	Competitive Programming		8	

NAME: PARVENDHAN C

REG NO: 241801197

DIVIDE AND CONQUER PROGRAM

1:

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 find_first_zero(int arr[], int low, int high) {
    int result = -1;
    while (low <= high) {
        int mid = (low + high) / 2;
        if (arr[mid] == 0) {
            result = mid;
            high = mid - 1;
        } else {
            low = mid + 1;
        }
    }
    return result;
}

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

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

    int first_zero_index = find_first_zero(arr, 0, m - 1);

    if (first_zero_index == -1) {
        printf("0\n");
    } else {
        printf("%d",m - first_zero_index);
    }

    return 0;
}

```

	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 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 0 0	2	2	✔

PROGRAM 2:

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 `⌊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 * 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
#include <stdio.h>

int majorityElement(int nums[], int numsSize) {
    int count = 0, candidate = 0;

    for (int i = 0; i < numsSize; i++) {
        if (count == 0) {
            candidate = nums[i];
            count = 1;
        } else if (nums[i] == candidate) {
            count++;
        } else {
            count--;
        }
    }
    return candidate;
}

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

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

    int result = majorityElement(nums, n);
    printf("%d\n", result);

    return 0;
}
```

	Input	Expected	Got	
✔	3 3 2 3	3	3	✔

Passed all tests! ✔

PROGRAM 3:

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 %)

```c
#include <stdio.h>

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

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

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

    return floor;
}

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

    return 0;
}
```

	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! ✔

PROGRAM 4:

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 elem1 = 0, elem2 = 0;

int findPair(int arr[], int left, int right, int x) {
    if (left >= right) {
        return 0;
    }

    int sum = arr[left] + arr[right];
    if (sum == x) {
        elem1 = arr[left];
        elem2 = arr[right];
        return 1;
    }
    else if (sum < x) {
        return findPair(arr, left + 1, right, x);
    }
    else {
        return findPair(arr, left, right - 1, x);
    }
}

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);

    if (findPair(arr, 0, n - 1, x)) {
        printf("%d\n%d\n", elem1, elem2);
    } else {
        printf("No\n");
    }

    return 0;
}
```

	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.

PROGRAM 5:

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
#include <stdio.h>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = low - 1;

    for (int j = low; j <= high - 1; j++) {
        if (arr[j] <= pivot) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return i + 1;
}

void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);

        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

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

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

    quickSort(arr, 0, n - 1);

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

    return 0;
}
```

	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.