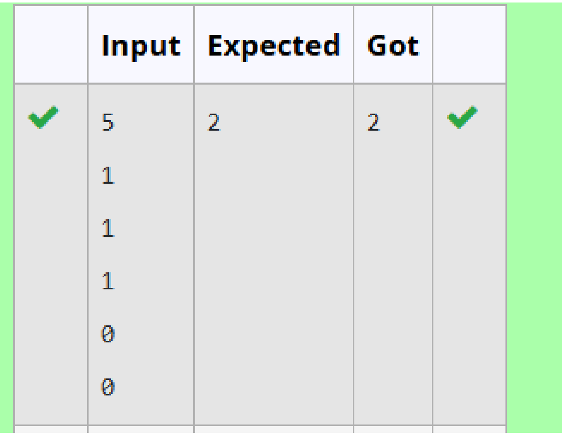
Name:Nandana Pramod
Roll no: 241801180
Exp no:2

Program 1:

```
#include <stdio.h>
int findFirstZero(int arr[], int low, int high)
{
    if (high >= low)
    {
        int mid = (low + high) / 2;
        if ((mid == 0 || arr[mid - 1] == 1) && arr[mid] == 0)
            return mid;
        if (arr[mid] == 1)
            return findFirstZero(arr, mid + 1, high);
        return findFirstZero(arr, low, mid - 1);
    }
    return -1;
}
int countZeros(int arr[], int m)
{
    int firstZeroIndex = findFirstZero(arr, 0, m - 1);
    if (firstZeroIndex == -1)
        return 0;
    return m - firstZeroIndex;
}
int main()
{
    int m;
    scanf("%d", &m);
    int arr[m];
    for (int i = 0; i < m; i++)
    {
        scanf("%d", &arr[i]);
    }
    printf("%d\n", countZeros(arr, m));
    return 0;
}
```

Output:

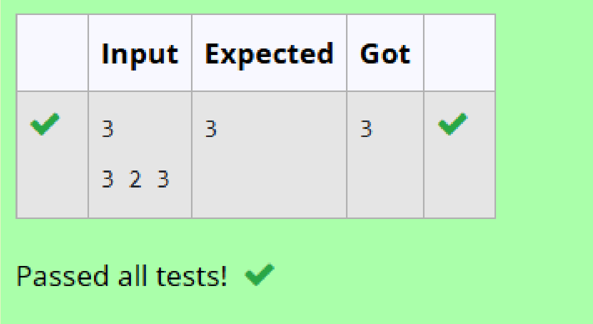


	Input	Expected	Got	
✓	5 1 1 1 0 0	2	2	✓

Program 2:

```
#include <stdio.h>
int majorityElement(int nums[], int n)
{
    for (int i = 0; i < n; i++)
    {
        int count = 0;
        for (int j = 0; j < n; j++)
        {
            if (nums[j] == nums[i])
            {
                count++;
            }
        }
        if (count > n / 2) {
            return nums[i];
        }
    }
    return -1;
}
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;
}
```

Output:



	Input	Expected	Got	
✓	3 3 2 3	3	3	✓

Passed all tests! ✓

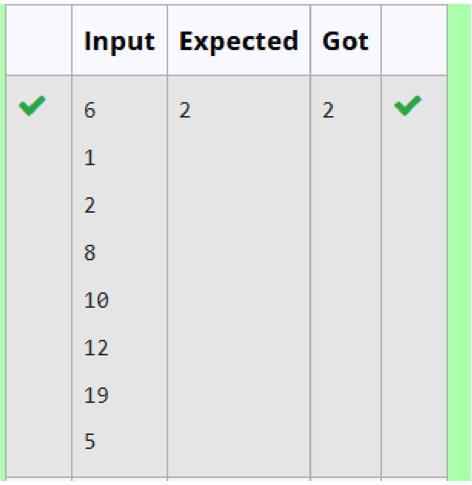
Program 3:

```
#include <stdio.h>
int findFloor(int arr[], int n, int x)
{
    int left = 0, right = n - 1;
    int floorValue = -1;
    while (left <= right)
    {
        int mid = left + (right - left) / 2;

        if (arr[mid] == x)
        {
            return arr[mid];
        }

        if (arr[mid] < x)
        {
            floorValue = arr[mid];
            left = mid + 1;
        } else
        {
            right = mid - 1;
        }
    }
    return floorValue;
}
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);
    if (result == -1)
    {
        printf("No floor found\n");
    }
    else
    {
        printf("%d\n", result);
    }
    return 0;
}
```

Output:



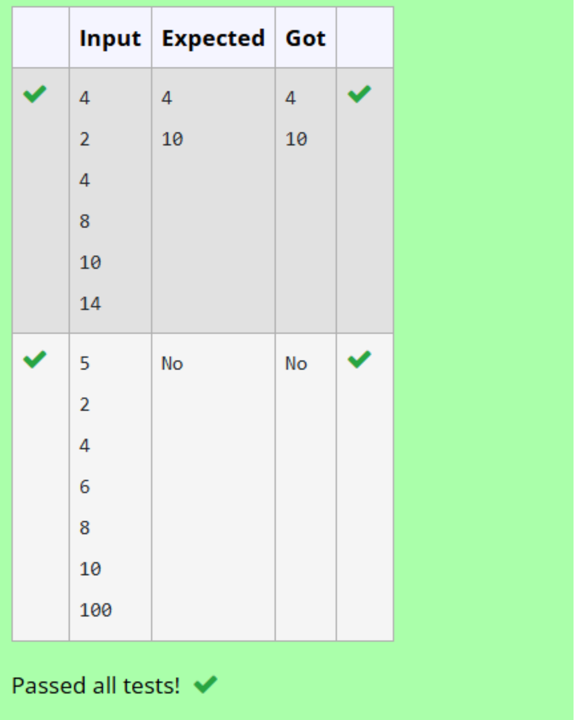
	Input	Expected	Got	
✓	6 1 2 8 10 12 19 5	2	2	✓

Program 4:

```
#include <stdio.h>
int findPair(int arr[], int left, int right, int x)
{
    if (left >= right)
    {
        return 0;
    }
    int sum = arr[left] + arr[right];
    if (sum == x)
    {
        printf("%d\n", arr[left]);
        printf("%d\n", arr[right]);
        return 1;
    }
    if (sum < x)
    {
        return findPair(arr, left + 1, right, x);
    }
    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("No\n");
    }
    return 0;
}
```

Output:



	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! ✓

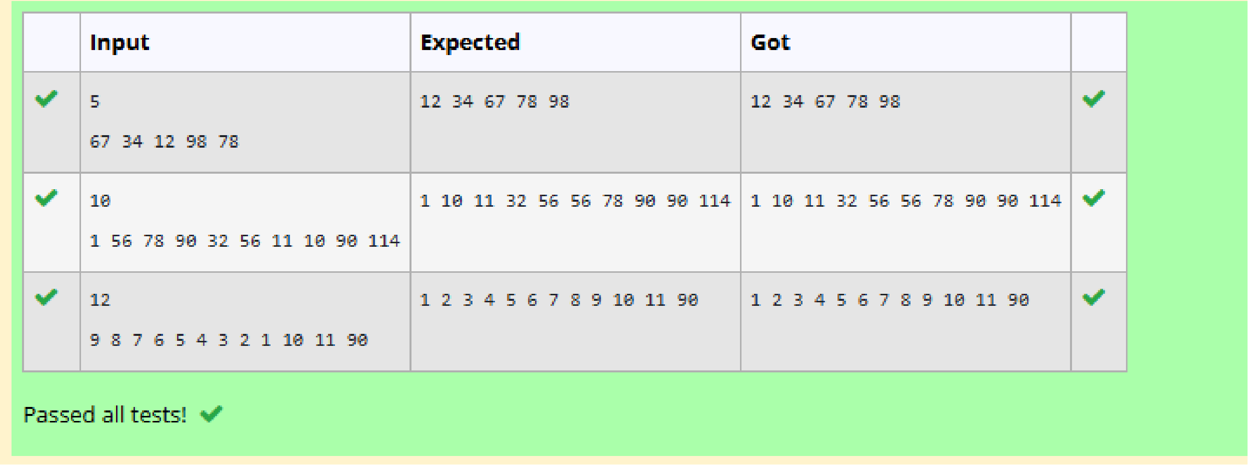
Program 5:

```
#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; 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;
}
```

Output:



	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! ✓