**DATA STRUCTURE (KCS -301)**

**LAB PRACTICAL FILE**

**NAME :SHREYA YADAV**

**ROLL NO. : 1901610100198**

**PROGRAM: *1******IMPLEMENTATION OF TRAVERSE***

**INPUT:**

#include<stdio.h>

int main()

{

int i,size,a[40];

printf("Enter the no. of elements of array\n");

scanf("%d", &size);

if (size > 40)

{

printf("Overflow");

}

else

{

printf("Elements of array :\n");

for ( i = 0; i < size; i++)

{

scanf("%d",&a[i]);

}

printf("Elements are : ");

for ( i = 0; i < size; i++)

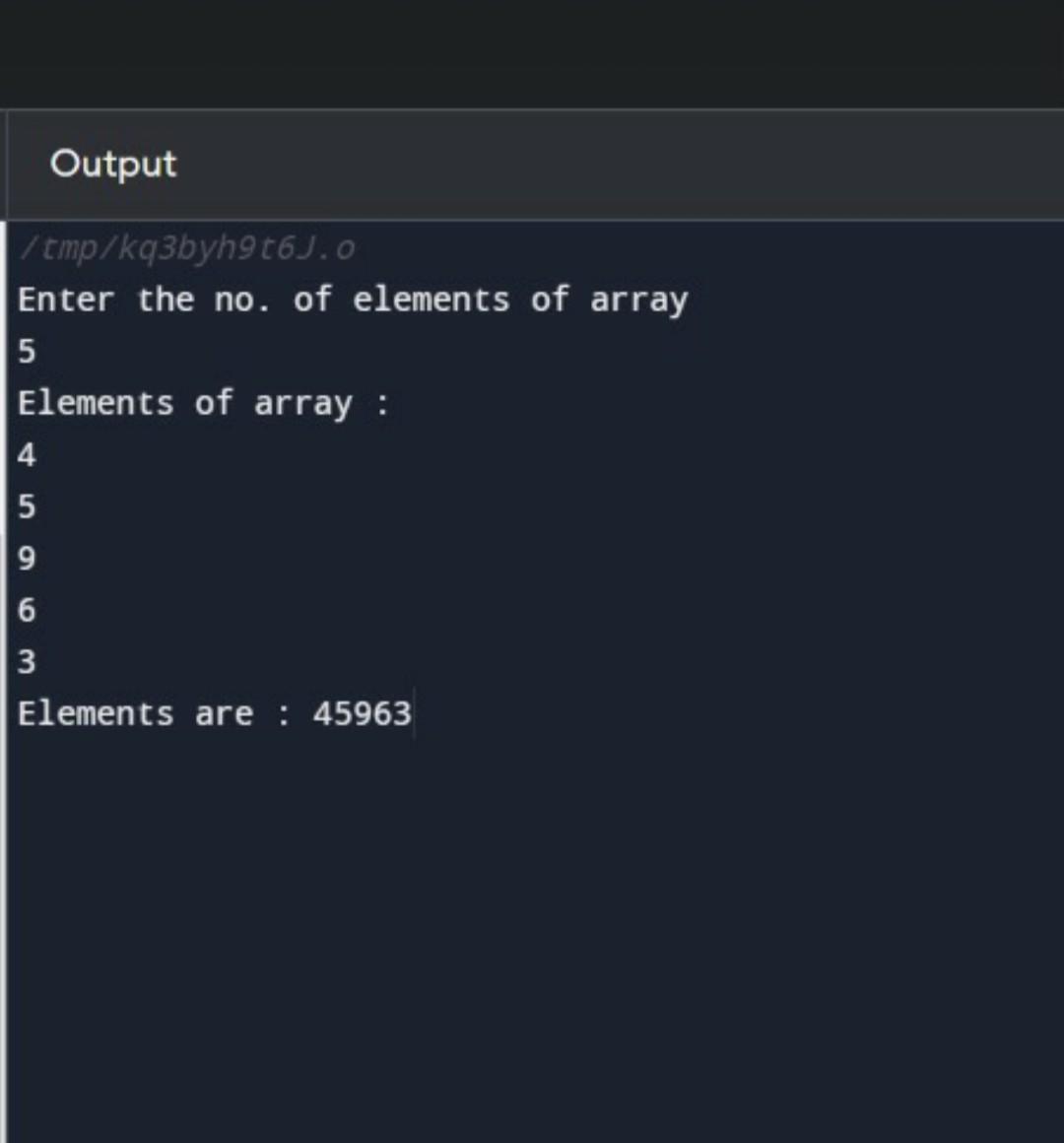
{

printf("%d", a[i]);

}

}

}

**OUTPUT :**

**PROGRAM : *2******IMPLEMENTATION OF BINARY SEARCH***

**INPUT:**

#include<stdio.h>

int main()

{

int arr[50],i,n,x,flag = 0,first,last,mid;

printf("enter the size of array:");

scanf("%d",&n);

printf("enter the elements of array (ascending order)\n");

for(i=0;i<n;++i)

{

scanf("%d", &arr[i]);

}

printf("enter the elements to search:\n");

scanf("%d",&x);

first=0;

last=n-1;

while(first <= last)

{

mid = first+last / 2;

if(x == arr[mid])

{

flag = 1;

break;

}

else if(x > arr[mid])

first = mid + 1;

else

last = mid - 1;

}

if(flag == 1)

printf("element is found at position %d",mid+1);

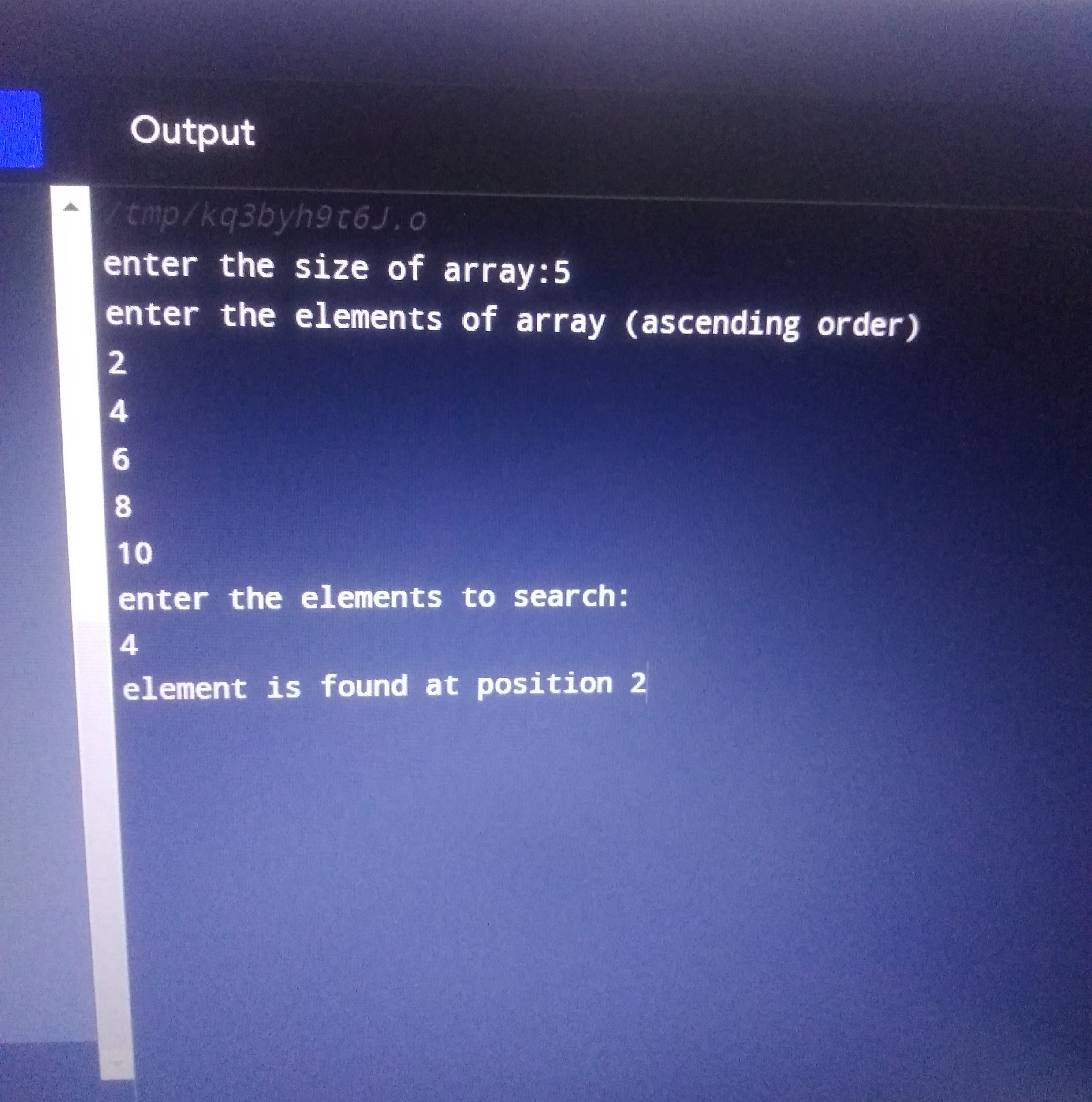
else

printf("elment not found");

return 0;

}

**OUTPUT :**



**PROGRAM : *3******IMPLEMENTATION OF LINEAR SEARCH***

**INPUT :**

include <stdio.h>

int main()

{

int a[20], i, x, n;

printf("How many elements? \n");

scanf("%d", &n);

printf("Enter array elements:\n ");

for (i = 0; i < n; ++i)

scanf("%d", &a[i]);

printf("\nEnter element to search: \n");

scanf("%d", &x);

for (i = 0; i < n; ++i)

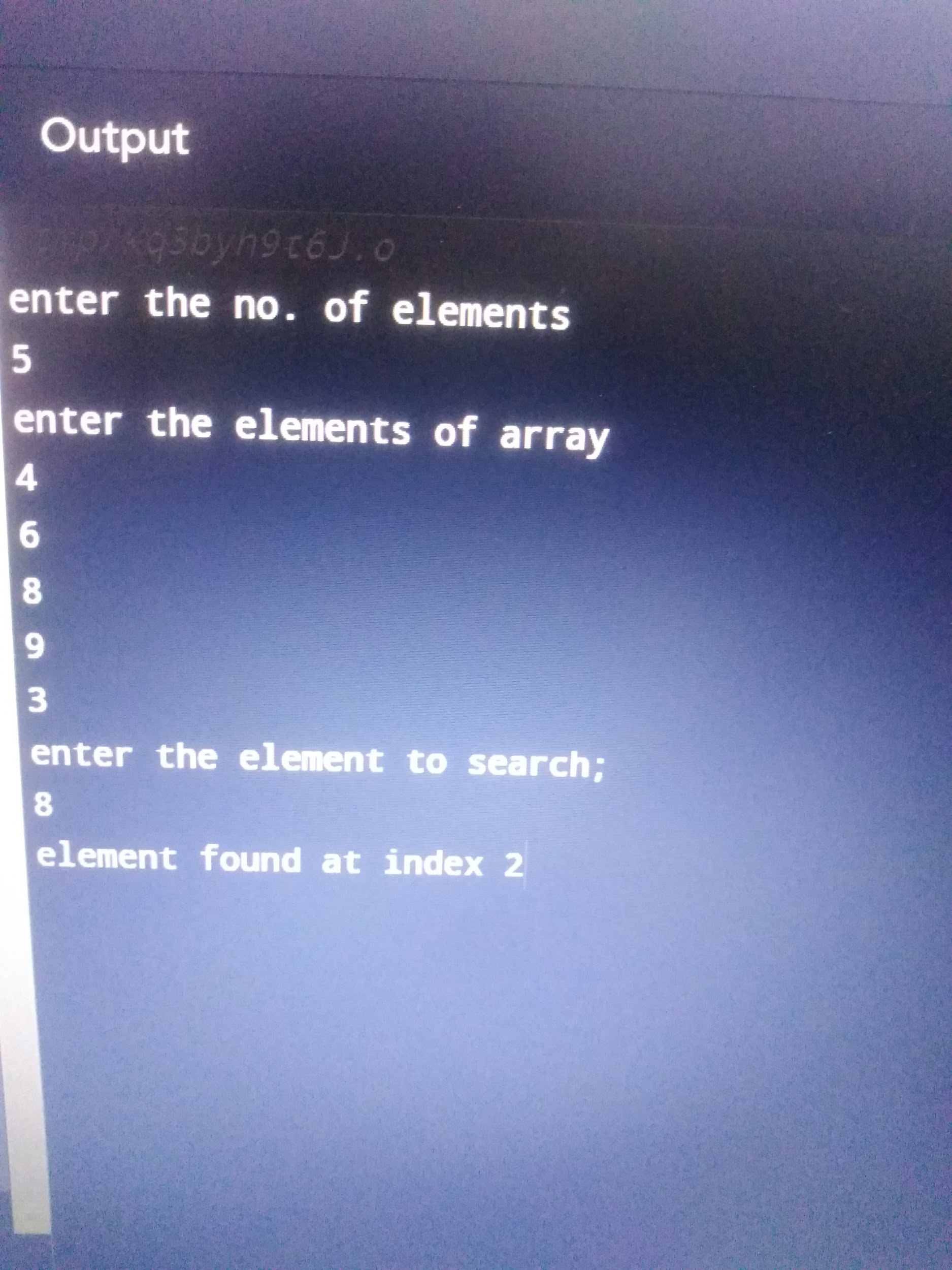
if (a[i] == x)

break;

if (i < n)

printf("Element found at index %d", i);

**OUTPUT :**

****

**PROGRAM : *4******IMPLEMENTATION OF INSERTION SORT***

**INPUT :**

int main()

{

int i, j, n, temp, a[30];

printf("Enter the number of elements:");

scanf("%d", &n);

printf("\nEnter the elements\n");

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

for (i = 1; i <= n - 1; i++)

{

temp = a[i];

j = i - 1;

while ((temp < a[j]) && (j >= 0))

{

a[j + 1] = a[j]; //moves element forward

j = j - 1;

}

a[j + 1] = temp; //insert element in proper place

}

printf("\nSorted list is as follows\n");

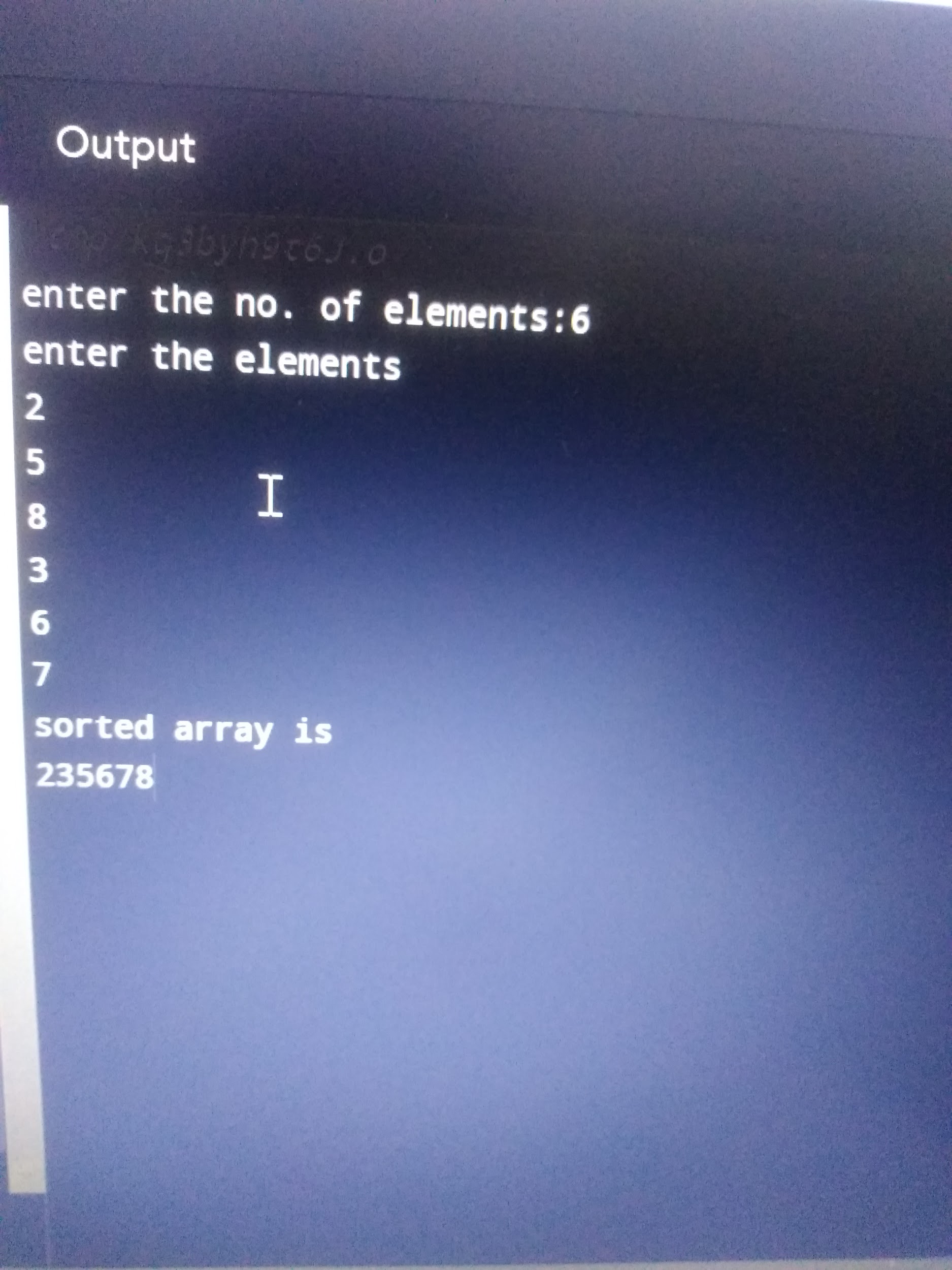
for (i = 0; i < n; i++)

{

printf("%d ", a[i]);

}

**OUTPUT :**



**PROGRAM : *5******IMPLEMENTATION OF SELECTION SORT***

**INPUT :**

#include <stdio.h>

int main()

{

int array[100], n, c, d, position, t;

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

for (c = 0; c < (n - 1); c++)

{

position = c;

for (d = c + 1; d < n; d++)

{

if (array[position] > array[d])

position = d;

}

if (position != c)

{

t = array[c];

array[c] = array[position];

array[position] = t;

}

}

printf("Sorted list in ascending order:\n");

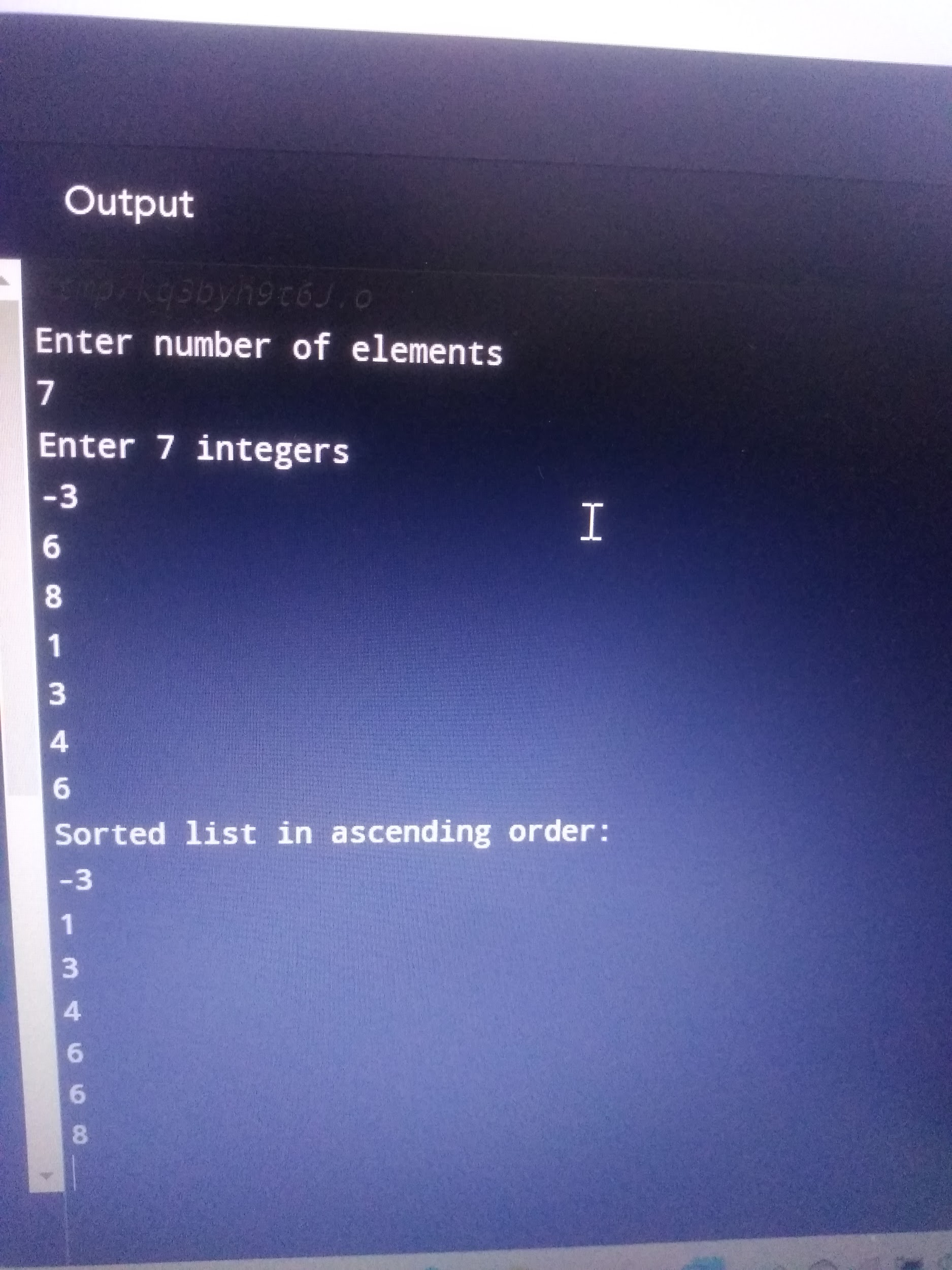
for (c = 0; c < n; c++)

printf("%d\n", array[c]);

return 0;

}

**OUTPUT :**



**PROGRAM : *6******IMPLEMENTATION OF BUBBLE SORT***

**INPUT :**

#include <stdio.h>

int main()

{

    int array[100], n, c, d, swap;

    printf("Enter number of elements\n");

    scanf("%d", &n);

    printf("Enter %d integers\n", n);

    for (c = 0; c < n; c++)

        scanf("%d", &array[c]);

    for (c = 0; c < n - 1; c++)

    {

        for (d = 0; d < n - c - 1; d++)

        {

            if (array[d] > array[d + 1]

            {

                swap = array[d];

                array[d] = array[d + 1];

                array[d + 1] = swap;

            }

        }

    }

    printf("Sorted list in ascending order:\n");

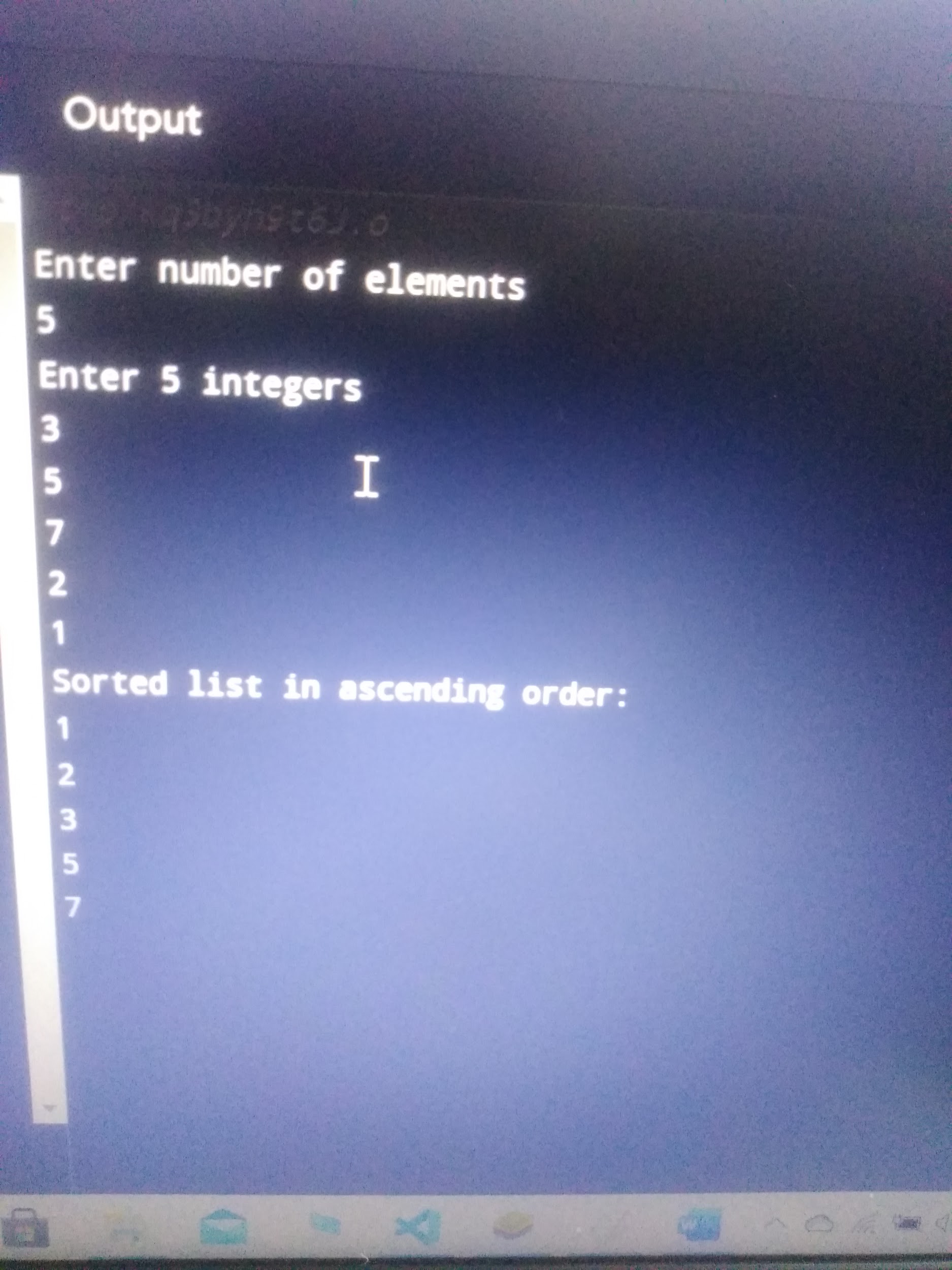
    for (c = 0; c < n; c++)

        printf("%d\n", array[c]);

    return 0;

}

**OUTPUT :**



**PROGRAM : *7******IMPLEMENTATION OF MERGE SORT***

**INPUT :**

#include <stdio.h>

void merge(int arr[], int p, int q, int r)

{

    int n1 = q - p + 1;

    int n2 = r - q;

    int L[n1], M[n2];

    for (int i = 0; i < n1; i++)

        L[i] = arr[p + i];

    for (int j = 0; j < n2; j++)

        M[j] = arr[q + 1 + j];

    int i, j, k;

    i = 0;

    j = 0;

    k = p;

    while (i < n1 && j < n2)

    {

        if (L[i] <= M[j])

        {

            arr[k] = L[i];

            i++;

        }

        else

        {

            arr[k] = M[j];

            j++;

        }

        k++;

    }

    while (i < n1)

    {

        arr[k] = L[i];

        i++;

        k++;

    }

    while (j < n2)

    {

        arr[k] = M[j];

        j++;

        k++;

    }

}

void mergeSort(int arr[], int l, int r)

{

    if (l < r)

    {

        int m = l + (r - l) / 2;

        mergeSort(arr, l, m);

        mergeSort(arr, m + 1, r);

        merge(arr, l, m, r);

    }

}

// Print the array

void printArray(int arr[], int size)

{

    for (int i = 0; i < size; i++)

        printf("%d ", arr[i]);

    printf("\n");

}

// Driver program

int main()

{

    int arr[] = {6, 5, 12, 10, 9, 1};

    int size = sizeof(arr) / sizeof(arr[0]);

    mergeSort(arr, 0, size - 1);

    printf("Sorted array using merge sort : \n");

    printArray(arr, size);

}

**OUTPUT :**

