DAY 11 – LINEAR AND BINARY SEARCH

13.. Write a menu driven C program to implement searching algorithms –

a. Linear search

b. Binary search.

PROGRAM

```c
#include<stdio.h>
#include<stdlib.h>
int arr[20],n;
void read(int x)
{
    int i;
    printf("Enter the elements:\n");
    for(i=0;i<x;i++)
        scanf("%d",&arr[i]);
}
void display()
{
    int i;
    for(i=0;i<n;i++)
        printf("%d ",arr[i]);
}
void linear(int x)
{
    int i,flag=0;
    for(i=0;i<n;i++)
    {
        if(x==arr[i])
        {
            printf("ELement found.");
            flag=1;
            break;
        }
    }
    if(flag==0)
        printf("\nElement not found.\n");
}
void binary(int x)
{
    int temp,beg=0,mid,end,i,j,flag=0;
    for(i = 0; i < n - 1; i++)
    {
        for(j = 0; j < n - i - 1; j++)
        {
            if(arr[j] > arr[j + 1])
            {
```

```
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j+1] = temp;
            }
        }
    }
    printf("Sorted array: ");
    display();
    end=n-1;
    while(beg<=end)
    {
        mid=(beg+end)/2;
        if(arr[mid]==x)
        {
            printf("\nelement found");
            flag=1;
            break;
        }
        else if(arr[mid]>x)
            end=mid-1;
        else
            beg=mid+1;
    }
    if(flag==0)
        printf("\nElement not found.\n");
}
int main()
{
    int ch,a;
    while(1)
    {
        printf("\nMENU:\n1.Enter the array.\n2.Linear search.\n3.Binary search.\n4.Exit.\nEnt
er your choice: ");
        scanf("%d",&ch);
        switch(ch)
        {
        case 1: printf("\nEnter the number of elements in array: ");
                scanf("%d",&n);
                read(n);
                break;
        case 2: printf("Enter the element to search: ");
                scanf("%d",&a);
                linear(a);
                break;
        case 3: printf("Enter the element to search: ");
                scanf("%d",&a);
                binary(a);
                break;
        case 4: exit(0);
                break;
        }
    }
    return 0;
}
```

OUTPUT

```
MENU:
1.Enter the array.
2.Linear search.
3.Binary search.
4.Exit.
Enter your choice: 1

Enter the number of elements in array: 10
Enter the elements:
4 2 6 19 23 5 65 23 1 43

MENU:
1.Enter the array.
2.Linear search.
3.Binary search.
4.Exit.
Enter your choice: 2
Enter the element to search: 6
ELement found.
MENU:
1.Enter the array.
2.Linear search.
3.Binary search.
4.Exit.
Enter your choice: 3
Enter the element to search: 4
Sorted array: 1 2 4 5 6 19 23 23 43 65
element found
MENU:
1.Enter the array.
2.Linear search.
3.Binary search.
4.Exit.
Enter your choice: 2
Enter the element to search: 65
ELement found.
MENU:
1.Enter the array.
2.Linear search.
3.Binary search.
4.Exit.
Enter your choice: 2
Enter the element to search: 98

Element not found.
```

DAY 11 – BUBBLE,INSERTION AND SELECTION SORT

14. Write a menu driven C program to implement the following sorting techniques

a. Bubble Sort

b. Insertion Sort

c. Selection Sort

PROGRAM

```c
#include<stdio.h>
#include<stdlib.h>
int i, j, n, temp, a[10];
void entry()
{
    printf("\nEnter no. of elements in the array: ");
    scanf("%d", &n);
    printf("\nEnter the elements: ");
    for(i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
void display()
{
    for(i = 0; i < n; i++)
        printf("%d ", a[i]);
}
void bubble_sort()
{
    int temp;
    for(i = 0; i < n; i++)
    {
        for(j = 0; j < n - i - 1; j++)
        {
            if(a[j] > a[j + 1])
            {
                temp = a[j];
                a[j] = a[j + 1];
                a[j + 1] = temp;
            }
        }
    }
    printf("\nArray after bubble sort: ");
    display();
```

```
}
void insertion_sort()
{
    for(i = 1; i < n; i++)
    {
        temp = a[i];
        j = i - 1;
        while((temp < a[j]) && (j >= 0))
        {
            a[j + 1] = a[j];
            j--;
        }
        a[j + 1] = temp;
    }
    printf("\nArray after insertion sort: ");
    display();
}
int smallest(int k)
{
    int pos = k, small = a[k], i;
    for(i = k + 1; i < n; i++)
    {
        if(a[i] < small)
        {
            small = a[i];
            pos = i;
        }
    }
    return pos;
}
void selection_sort()
{
    int k, pos;
    for(k = 0; k < n; k++)
    {
        pos = smallest(k);
        temp = a[k];
        a[k] = a[pos];
        a[pos] = temp;
    }
    printf("Array after selection sort: ");
    display();
}
int main()
{
    int ch;
    while(1)
    {
        printf("\nMENU\n");
        printf("1. Entry\n2. Bubble Sort\n3. Insertion Sort\n4. Selection Sort\n5. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &ch);
        switch(ch)
        {
```

```
            case 1: entry();
                    printf("Array:");
                    display();
                    break;
            case 2: bubble_sort();
                    break;
            case 3: insertion_sort();
                    break;
            case 4: selection_sort();
                    break;
            case 5: exit(0);
                    break;
        }
    }
    return 0;
}
```

OUTPUT

```
MENU
1. Entry
2. Bubble Sort
3. Insertion Sort
4. Selection Sort
5. Exit
Enter choice: 1

Enter no. of elements in the array: 5

Enter the elements: 88 43 56 12 34
Array:88 43 56 12 34
MENU
1. Entry
2. Bubble Sort
3. Insertion Sort
4. Selection Sort
5. Exit
Enter choice: 2

Array after bubble sort: 12 34 43 56 88
MENU
1. Entry
2. Bubble Sort
3. Insertion Sort
4. Selection Sort
5. Exit
Enter choice: 3

Array after insertion sort: 12 34 43 56 88
MENU
1. Entry
2. Bubble Sort
3. Insertion Sort
4. Selection Sort
5. Exit
Enter choice: 4
Array after selection sort: 12 34 43 56 88
```