ASSIGNMENT-1

COURSE CODE: CSA0390

1.

A)Traverse

program:

```
#include <stdio.h>
int main()
{
   int arr[]={10,20,30,40,50};
   int n=sizeof(arr)/sizeof(arr[0]);
   for (int i=0;i<n;i++)
   {
     printf("%d ",arr[i]);
   }
   return 0;
}
```

Output: 10 20 30 40 50

B) Search:

Program:

```
#include <stdio.h>
int main()
{
   int arr[] = {10,20,30,40,50};
   int n=sizeof(arr)/sizeof(arr[0]);
   int key=40,found=0;
   for(int i=0;i<n;i++)
```

```
    {
        if(arr[i]==key)
        {
            printf("Element %d found at index %d.\n",key, i);
            found=1;
            break;
        }
    }
    if(!found)
    {
        printf("Element %d not found in the array.\n",key);
}
    return 0;
}
```

Output: Element 40 found at index 3.

C)insert:

Program:

```
#include <stdio.h>
#define SIZE 10
int main()
{
    int arr[SIZE]={10,20,30,40,50};
    int n=5;
    int pos=3;
    int element=25;
    printf("Original array elements:\n");
    for(int i=0;i<n;i++)
```

```
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    for(int i=n-1;i>=pos;i--)
    {
        arr[i+1]=arr[i];
    }
    arr[pos]=element;
    n++;
    printf("Array elements after insertion:\n");
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    return 0;
}
```

Output: Original array elements:

10 20 30 40 50

Array elements after insertion:

10 20 30 25 40 50

D)delete

program:

```
#include <stdio.h>
#define SIZE 10
int main()
{
```

```
    int arr[SIZE]={10,20,30,40,50};
    int n=5;
    int pos=2;
    printf("Original array elements:\n");
    for (int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    for (int i=pos;i<n-1;i++)
    {
        arr[i]=arr[i+1];
    }
    n--;
    printf("Array elements after deletion:\n");
    for (int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    return 0;
}
```

Output:

Original array elements:

10 20 30 40 50

Array elements after deletion:

10 20 40 50

E)update:

Program:

```
#include <stdio.h>
#define SIZE 5
int main()
{
   int arr[SIZE]={10,20,30,40,50};
   int pos=2;
   int new_value=35;
   printf("Original array elements:\n");
   for(int i=0;i< SIZE;i++)
   {
      printf("%d ",arr[i]);
   }
   printf("\n");
   arr[pos]=new_value;
   printf("Array elements after update:\n");
   for(int i=0;i<SIZE;i++)
   {
      printf("%d ",arr[i]);
   }
   printf("\n");
   return 0;
}
```

Output:

Original array elements:

10 20 30 40 50

Array elements after update:

10 20 35 40 50

2. Writing a recursive function to calculate the factorial of a number.

```
#include <stdio.h>
unsigned long long factorial(int n)
{
   if(n==0||n==1)
   {
      return 1;
   }
   else
   {
      return n*factorial(n-1);
   }
}
int main()
{
   int number;
   printf("Enter a integer: ");
   scanf("%d",&number);
   if (number < 0) {
      printf("Factorial undefined for negative numbers.\n");
   }
   else
   {
      unsigned long long fact=factorial(number);
      printf("Factorial of %d=%llu\n",number,fact);
   }
   return 0;
}
```

Output:

```
Enter a integer: 6
Factorial of 6=720
```

3. Write a C Program to find duplicate element in an array

```
#include <stdio.h>
#define SIZE 10
int main()
{
   int arr[SIZE]={2,5,3,7,2,8,3,1,5,6};
   int seen[SIZE]={0};
   int duplicate_found=0;
   printf("Duplicate elements in the array: ");
   for (int i=0;i<SIZE;i++)
   {
      if(seen[arr[i]]==1)
      {
         printf("%d ",arr[i]);
         duplicate_found=1;
      }
      else
      {
         seen[arr[i]]=1;
      }
   }
   if (!duplicate_found)
   {
      printf("No duplicates found.");
   }

   printf("\n");

   return 0;
}
```

Output: Duplicate elements in the array: 2 3 5

4. Write a small C Program to find Max and Min from an array elements

```
#include <stdio.h>
#define SIZE 10
int main()
{
   int arr[SIZE]={2,5,3,7,1,8,6,4,9,10};
   int max=arr[0];
   int min=arr[0];
   for (int i=1;i<SIZE;i++)
   {
      if(arr[i]>max)
      {
         max=arr[i];
      }
      if(arr[i]<min)
      {
         min=arr[i];
      }
   }
   printf("Maximum element in the array: %d\n",max);
   printf("Minimum element in the array: %d\n",min);
   return 0;
}
```

Output: Maximum element in the array: 10

Minimum element in the array: 1

5. Given a number n. the task is to print the Fibonacci series and the sum of the series using recursion.

input: n=10

output: Fibonacci series

0, 1, 1, 2, 3, 5, 8, 13, 21, 34

Sum: 88

Program:

```
#include <stdio.h>
unsigned long long fibonacci(int n,unsigned long long*sum)
{
   if (n<=1)
   {
      *sum+=n;
      return n;
   }
   else
   {
      unsigned long long fib_n=fibonacci(n-1,sum)+fibonacci(n-2,sum);
      *sum+=fib_n;
      return fib_n;
   }
}
int main()
{
   int n=10;
   unsigned long long sum=0;
   printf("Fibonacci series:\n");
   for (int i=0;i<n;i++)
   {
      unsigned long long fib=fibonacci(i,&sum);
```

```
        printf("%llu", fib);
        if (i<n-1)
        {
            printf(", ");
        }
    }
    printf("\n");
    printf("Sum: %llu\n",sum);
    return 0;
}
```

Output:

Fibonacci series:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34

Sum: 512

6. You are given an array arr in increasing order. Find the element x from arr using binary search.

Example 1: arr={ 1,5,6,7,9,10},X=6

Output : Element found at location 2

Example 2: arr={ 1,5,6,7,9,10},X=11

Output : Element not found at location 2

```
#include <stdio.h>
int binarySearch(int arr[],int left,int right,int x)
{
    while(left<=right)
    {
        int mid=left+(right-left)/2;
        if (arr[mid]==x)
        {
```

```
            return mid;
        }
        else if(arr[mid]<x)
        {
            left=mid+1;
        }
        else
        {
            right=mid-1;
        }
    }
    return -1;
}
int main()
{
    int arr[]={1,5,6,7,9,10};
    int n=sizeof(arr)/sizeof(arr[0]);
    int x=6;
    int result=binarySearch(arr,0,n-1,x);
    if (result!=-1)
    {
        printf("Element found at location %d\n",result);
    }
    else
    {
        printf("Element not found in the array.\n");
    }
    return 0;
}
```

Output: Element found at location 2

7.write a code for linear search in c programming

```
#include <stdio.h>
int linearSearch(int arr[],int size,int target)
{
    for(int i=0;i<size;i++)
    {
        if(arr[i]==target)
        {
            return i;


        }
    }
    return -1;
}
int main()
{
    int arr[]={2,4,6,8,10,12,14,16};
    int size=sizeof(arr)/sizeof(arr[0]);
    int target=10;
    int result=linearSearch(arr, size,target);


    if(result!=-1)
    {
        printf("Element found at index: %d\n",result);
    }
    else
    {
        printf("Element not found in the array.\n");
    }
    return 0;
```

```
}
```

Output:Element found at index:4

8.Write a program for binary search in c programming

```
#include <stdio.h>
int binarySearch(int arr[],int size,int target)
{
   int left=0;
   int right=size-1;
   while(left<=right)
   {
      int mid=left+(right-left)/2;
      if(arr[mid]==target)
      {
         return mid;
      }
      if(arr[mid]<target)
      {
         left=mid+1;
      }
      else
      {
         right=mid-1;
      }
   }
   return -1;
}

int main()
```

```
{
    int arr[]={2,4,6,8,10,12,14,16};
    int size=sizeof(arr)/sizeof(arr[0]);
    int target=10;
    int result=binarySearch(arr,size,target);

    if (result!=-1)
    {
        printf("Element found at index: %d\n",result);
    }
    else
    {
        printf("Element not found in the array.\n");
    }
    return 0;
}
```

Output:Element found at index:4