

## 1. Write a C Program to implement following operations

### a) Traverse

```
#include <stdio.h>
#define MAX_SIZE 100
int main() {
    int arr[MAX_SIZE]; int n; printf("Enter number of
    elements in array: "); scanf("%d", &n);
    printf("Enter %d elements:\n", n); for (int i = 0; i < n;
    i++) {
        scanf("%d", &arr[i]);
    }
    printf("Array elements: "); for
    (int i = 0; i < n; i++) { printf("%d",
    arr[i]);
    } printf("\n");
    return 0;
}
```

### OUTPUT:

Enter number of elements in array: 5

Enter 5 elements: 1 2 5 8 4

Array elements: 1 2 5 8 4

### b) Search

```
#include <stdio.h>
#define SIZE 5
int main() {
    int arr[SIZE] = {1, 2, 3, 4, 5}; int
    searchElement = 9; int i, found
    = 0; for (i = 0; i < SIZE; i++) {
        if (arr[i] == searchElement) {
            found = 1; break;
        }
    }
    if (found) {
        printf("Element %d found in the array.\n", searchElement);
    } else { printf("Element %d not found in the array.\n", searchElement);
    }
    return 0;
}
```

## OUTPUT:

Element 9 not found in the array.

### c) Insert

```
#include <stdio.h>
#define SIZE 5
int
main() {
    int arr[SIZE] = {1, 2, 4, 5};
    int insertIndex = 2;
    int newValue = 3;
    int i;
    printf("Initial Array: ");
    for (i = 0; i < SIZE; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    for (i = SIZE - 1; i > insertIndex; i--) {
        arr[i] = arr[i - 1];
    }
    arr[insertIndex] = newValue;
    printf("Array after Insertion: ");
    for (i = 0; i < SIZE; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

## OUTPUT:

Initial Array: 1 2 4 5 0

Array after Insertion: 1 2 3 4 5

### d) Delete

```
#include <stdio.h>
#define SIZE 5
int
main() {
    int arr[SIZE] = {1, 2, 3, 4, 5};
    int deleteIndex = 2;
    int i;
    printf("Initial Array: ");
    for (i = 0; i < SIZE; i++) {
        printf("%d ", arr[i]);
    }
}
```

```

    }printf("\n");
    for (i=deleteIndex; i<SIZE-1; i++) {arr[i]=arr[i+1];
    }
    printf("Array after Deletion: "); for (i
    =0; i<SIZE-1; i++) {printf("%d ",
    arr[i]);
    }
}

```

### OUTPUT:

Initial Array: 1 2 3 4 5

Array after Deletion: 1 2 4 5

### e) Update

```

#include <stdio.h>
#define SIZE 5
int main(){
    int arr[SIZE]={1,2,3,4,5}; int
    updateIndex=2; int newValue
    =10;
    int i;
    printf("Initial Array: ");
    for (i=0; i<SIZE; i++) {printf("%d ",
    arr[i]);
    }printf("\n");
    arr[updateIndex]=newValue;
    printf("Array after Update: "); for (i
    =0; i<SIZE; i++) {printf("%d ",
    arr[i]);
    }printf("\n");
    return 0;
}

```

### OUTPUT:

Initial Array: 1 2 3 4 5

Array after Update: 1 2 10 4 5

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

### PROGRAM:

```

#include <stdio.h> int
factorial(int n); int
main() {
    int num; printf("Enter a
    number: "); scanf("%d",
    &num);
    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    } else { int result =
        factorial(num);
        printf("Factorial of %d = %d\n", num, result);
    }
    return 0;
} int factorial(int n) {
    if (n == 0) {
        return 1;
    }
    else {
        return n * factorial(n - 1);
    }
}

```

## OUTPUT

Enter a number : 5  
 Factorial of 5 = 120

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

```

#include <stdio.h>
#define MAX_SIZE 100 int
main() {
    int arr[MAX_SIZE];
    int n;
    printf("Enter number of elements in array: ");
    scanf("%d", &n); printf("Enter %d elements:\n", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Duplicate elements in the array are: ");
}

```

```

for(int i=0;i<n;i++){for(int j=i
+1;j<n;j++){//If duplicate
found if(arr[i]==arr[j]){
    printf("%d",arr[i]);break;
}
}
}printf("\n");
return 0;
}

```

### OUTPUT:

Enter number of elements in array: 8 Enter 8  
elements:

1 2 5 6 6 8 4 2

Duplicate elements in the array are: 2 6

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

### PROGRAM:

```

#include<stdio.h>
#define MAX_SIZE 100
int main() {
    int arr[MAX_SIZE];
    int n; int max,
    min;
    printf("Enter number of elements in array: ");
    scanf("%d", &n); printf("Enter %d elements:\n", n);
    for (int i = 0; i < n; i++) { scanf("%d", &arr[i]);
    }
    max = arr[0]; min = arr[0];
    for (int i = 1; i < n; 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:

Enter number of elements in array: 6 Enter 6  
elements:

1 5 3 7 8 12

Maximum element in the array: 12

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>  
int main() { int n;  
    printf("Enter the number of terms in Fibonacci series: ");  
    scanf("%d", &n); int fib[n];  
    int sum = 0; if  
    (n >= 1) { fib[0]  
    = 0;  
        sum += fib[0];  
        printf("%d", fib[0]);  
    }  
    if (n >= 2) { fib[1] =  
        1;  
        sum += fib[1];  
        printf("%d", fib[1]);  
    }  
    for (int i = 2; i < n; i++) {
```

```

        fib[i] = fib[i-1] + fib[i-2];
        sum += fib[i]; printf("%d ",
        fib[i]);
    }
    printf("\nSum of Fibonacci series: %d\n", sum);

    return 0;
}

```

### OUTPUT:

Enter the number of terms in Fibonacci series: 10  
 0 1 1 2 3 5 8 13 21 34  
 Sum of Fibonacci series: 88

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

### PROGRAM:

```

#include <stdio.h>
#define MAX_SIZE 100
int main() { int
arr[MAX_SIZE];
    int n, x; int found =
    0; int location =
    -1;
    printf("Enter number of elements in array: "); scanf("%d", &n);
    printf("Enter %d elements in increasing order:\n", n);
    for (int i = 0; i < n; i++) { scanf("%d",
        &arr[i]);
    }
    printf("Enter element to search: ");
    scanf("%d", &x); int low = 0; int high = n
    - 1; while (low <= high) { int mid = (low
    + high) / 2;

```



```

        if(arr[mid]==x){
            found = 1;location
            =mid;break;
        }else if (arr[mid]<x){low =
            mid+ 1;
        }else{high = mid-
            1;
        }
    }
    if(found){
        printf("Element %d found at location %d.\n",x,location);
    }else{printf("Element %d not found.\n",
        x);
    }

    return 0;
}

```

### OUTPUT:

Enter number of elements in array: 8 Enter 8  
 elements in increasing order:  
 1 2 8 12 14 16 18 20  
 Enter element to search: 12 Element 12  
 found at location 3.

## 7. Write C Program Find the element x from the array using Linear search.

### PROGRAM:

```

#include <stdio.h>
#define MAX_SIZE 100
int main(){int
arr[MAX_SIZE];
    int n,x;int
    found=0;
    printf("Enter number of elements in array: ");scanf("%d",&n);
    printf("Enter %d elements:\n",n);for
    (inti=0;i<n;i++){scanf("%d",&arr[i]);
    }
    printf("Enter element to search: ");
    scanf("%d",&x);for(inti=0;i<n;i++){
    if(arr[i]==x){
        found = 1;
    }
    }
}

```



```
        printf("Element %d found at location %d.\n", x, i); break;
    }
}
if(!found){
    printf("Element %d not found in the array.\n", x);
}
return 0;
}
```

### **OUTPUT:**

*Enter number of elements in array: 6 Enter 6 elements:*

*1 2 9 4 7 6*

*Enter element to search: 6*

*Element 6 found at location 5.*

