DATA STRUCTURE

DATE:24/07/24

1) write a C program to implement following operations

a)traverse:

PROGRAM:

```
#include<stdio.h>
int main() {
   int array[] = {1, 2, 3, 4, 5};
  int size = sizeof(array) / sizeof(array[0]);
   printf("Elements of the array: ");
for (int i = 0; i < size; ++i) {
      printf("%d ", array[i]);
}
   printf("\n");
   return 0;
}
```

OUTPUT:

Elements of the array: 1 2 3 4 5

b)search

PROGRAM:

```
#include <stdio.h>
int linearSearch(int arr[], int size, int key) {
for (int i = 0; i < size; ++i) {
      if (arr[i] == key) {
return i;
      }
}
   return -1;
```

```
}
int main() {
int array[] = {10, 20, 30, 40, 50};
   int size = sizeof(array) / sizeof(array[0]);
int key = 30;
   int index = linearSearch(array, size, key);
if (index != -1) {
      printf("Element %d found at index %d\n", key, index);
} else {
      printf("Element %d not found in the array\n", key);
}
return 0;
}
```

OUTPUT:

Element 30 found at index 2

C)delete:

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int deleteElement(int arr[], int size, int index) {
   if (index < 0 || index >= size) {
      printf("Invalid index to delete\n");
      return size;
   }
   for (int i = index; i < size - 1; ++i) {
      arr[i] = arr[i + 1];
   }
return size - 1;
```

```
}
int main() {
   int array[MAX_SIZE] = {10, 20, 30, 40, 50};
   int size = 5;
   int index = 2;
printf("Array before deletion:\n");
   for (int i = 0; i < size; ++i) {
      printf("%d ", array[i]);
   }
   printf("\n");
   size = deleteElement(array, size, index);
printf("Array after deletion:\n");
   for (int i = 0; i < size; ++i) {
      printf("%d ", array[i]);
   }
   printf("\n");
return 0;
}
```

OUTPUT:

Array before deletion:

10 20 30 40 50

Array after deletion:

10 20 40 50

d)update:

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
void updateElement(int arr[], int size, int index, int newValue) {
```

```
    if (index < 0 || index >= size) {
        printf("Invalid index to update\n");
        return;
    }
arr[index] = newValue;
}
int main() {
    int array[MAX_SIZE] = {10, 20, 30, 40, 50};
    int size = 5;
    int index = 2;
    int newValue = 35;
    printf("Array before update:\n");
    for (int i = 0; i < size; ++i) {
        printf("%d ", array[i]);
    }
    printf("\n");
updateElement(array, size, index, newValue);
printf("Array after update:\n");
    for (int i = 0; i < size; ++i) {
        printf("%d ", array[i]);
    }
    printf("\n");
return 0;
}
```

OUTPUT:

Array before update:

10 20 30 40 50

Array after update:

10 20 35 40 50

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

PROGRAM:

```
#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 non-negative integer: ");
   scanf("%d", &number);
   if (number < 0) {
      printf("Factorial is not defined for negative numbers.\n");
   } else {
      unsigned long long result = factorial(number);
      printf("Factorial of %d is: %llu\n", number, result);
   }
return 0;
}
```

OUTPUT:

Factorial of 5 is: 120

3)write a C program to find duplicate element in an array

PROGRAM:

```
#include <stdio.h>
```

```
#define MAX_SIZE 100
int main() {
   int array[MAX_SIZE];
   int size, i, j;
      printf("Enter size of the array: ");
   scanf("%d", &size);

   // Input elements of the array
   printf("Enter elements in the array:\n");
   for (i = 0; i < size; i++) {
      scanf("%d", &array[i]);
   }
    printf("Duplicate elements in the array are: ");
   for (i = 0; i < size; i++) {
      for (j = i + 1; j < size; j++) {
         if (array[i] == array[j]) {
            printf("%d ", array[i]);
            break;
         }
      }
   }
   printf("\n");
return 0;
}
```

OUTPUT:

Enter size of the array: 4

Enter elements in the array:

2 3 4 2

Duplicate elements in the array are: 2

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 array[MAX_SIZE];
   int size, i;
   int max, min;
      printf("Enter size of the array: ");
   scanf("%d", &size);
      printf("Enter elements in the array:\n");
   for (i = 0; i < size; i++) {
      scanf("%d", &array[i]);
   }
     max = min = array[0];
      for (i = 1; i < size; i++) {
      if (array[i] > max) {
         max = array[i];
      }
      if (array[i] < min) {
         min = array[i];
      }
   }
   printf("Maximum element in the array is: %d\n", max);
   printf("Minimum element in the array is: %d\n", min);
return 0;
}
```

OUTPUT:

Enter size of the array: 4

Enter elements in the array:4 2 3 1

Maximum element in the array is:4

Minimum element in the array is: 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 fibonacci(int n) {
   if (n <= 1)
      return n;
   return fibonacci(n - 1) + fibonacci(n - 2);
}
int main() {
   int n = 10;
   int sum = 0;
printf("Fibonacci Series:\n");
   for (int i = 0; i < n; i++) {
      printf("%d, ", fibonacci(i));
      sum += fibonacci(i);
   }
printf("\nSum: %d\n", sum);
return 0;
```

}

OUTPUT:

Fibonacci Series:

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

sum:88

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 locaion 2

PROGRAM:

```
#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;
       if (arr[mid] < x)
         left = mid + 1;
      else
         right = mid - 1;
   }
return -1;
}
int main() {
   int arr[] = {1, 5, 6, 7, 9, 10};
   int x = 6;
```

```
   int n = sizeof(arr) / sizeof(arr[0]);
   int result = binarySearch(arr, 0, n - 1, x);
if (result != -1)
      printf("Element found at location %d\n", result);
   else
      printf("Element not found\n");
return 0;
}
```

7)Linear search

PROGRAM:

```
#include <stdio.h>
int linearSearch(int arr[], int n, int x) {
   for (int i = 0; i < n; i++) {
      if (arr[i] == x) {
         return i;
      }
   }
   return -1;
}
int main() {
   int arr[] = {2, 5, 7, 9, 11};
   int n = sizeof(arr) / sizeof(arr[0]);
   int x = 7;
   int result = linearSearch(arr, n, x);
   if (result == -1) {
      printf("Element not found");
   } else {
      printf("Element found at index: %d", result);
```

```
    }
    return 0;
}
```

OUTPUT:Element found at index: 2

8)Binary search

PROGRAM:

```
#include <stdio.h>
int binarySearch(int arr[], int l, int r, int x) {
    while (l <= r) {
        int mid = l + (r - l) / 2;
if (arr[mid] == x)
            return mid;
  if (arr[mid] < x)
            l = mid + 1;
        else
            r = mid - 1;
    }
return -1;
}
int main() {
    int arr[] = {2, 3, 4, 10, 40};
    int n = sizeof(arr) / sizeof(arr[0]);
    int x = 10;
    int result = binarySearch(arr, 0, n - 1, x);
    if (result == -1)
        printf("Element not present in the array");
    else
        printf("Element found at index %d", result);
```

```
    return 0;
}
```

OUTPUT:Element found at index 3