DATA STRUCTURE

DAY 1 – 24/07/2024

1.write a c program to implement following operations

a. Traverse

PRAOGRAM:

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

OUTPUT:

```
Elements of the array:
1 2 3 4 5
```

b.serach

PROGRAM:

```
#include <stdio.h>
int main()
 {
   int arr[5] = {2, 4, 6, 8, 10};
   int target;
```

```
    int found = 0;
    printf("Enter the element to search for: ");
    scanf("%d", &target);
    for (int i = 0; i < 5; i++)
{
        if (arr[i] == target)
{
            found = 1;
            printf("Element %d found at index %d.\n", target, i);
            break;
        }
    }
    if (!found)
{
        printf("Element %d not found in the array.\n", target);
    }
    return 0;
}
```

OUTPUT:

Enter the element to search for: 6

Element 6 found at index 2.

c.insert

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
 {
    int arr[MAX_SIZE] = {1, 2, 3, 4, 5};
    int size = 5;
```

```
    int insertPos = 2;
    int newValue = 10;
    printf("Original Array:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n");
    for (int i = size - 1; i >= insertPos; i--)
{
        arr[i + 1] = arr[i];
    }
    arr[insertPos] = newValue;
    size++;
    printf("Array after insertion:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

OUTPUT:

Original Array:

1 2 3 4 5

Array after insertion:

1 2 10 3 4 5

d.delete

PROGRAM:

```
#include <stdio.h>
 #define MAX_SIZE 100
int main()
{
    int arr[MAX_SIZE] = {1, 2, 3, 4, 5};
    int size = 5;
    int deletePos = 2;
    printf("Original Array:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n");
    for (int i = deletePos; i < size - 1; i++)
{
        arr[i] = arr[i + 1];
    }
    size--;
    printf("Array after deletion:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

OUTPUT:

```
Original Array:
1 2 3 4 5
```

Array after deletion:

1 2 4 5

e.update

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
    int arr[MAX_SIZE] = {1, 2, 3, 4, 5};
    int size = 5;
    int updatePos = 2;
    int newValue = 10;
    printf("Original Array:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n");
    arr[updatePos] = newValue;
    printf("Array after update:\n");
    for (int i = 0; i < size; i++)
{
        printf("%d ", arr[i]);
    }
    printf("\n")
    return 0;
}
```

OUTPUT:

Original 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>
unsigned long long factorial(int n)
{
    if (n == 0)
{
        return 1;
    } else
{
        return n * factorial(n - 1);
    }
}
int main()
{
    int num;
    unsigned long long fact;
    printf("Enter a non-negative integer: ");
    scanf("%d", &num);
    fact = factorial(num);
    printf("Factorial of %d = %llu\n", num, fact);
    return 0;
}
```

OUTPUT:

Enter a non-negative integer: 5

Factorial of 5 = 120

3.Write a c program to find duplicate element in an aray

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int main()
{
   int arr[MAX_SIZE];
   int size;
   printf("Enter the size of the array (max %d): ", MAX_SIZE);
   scanf("%d", &size);
   printf("Enter elements of the array:\n");
   for (int i = 0; i < size; i++)
{
      scanf("%d", &arr[i]);
   }
   printf("Duplicate elements in the array are: ");
   for (int i = 0; i < size; i++)
{
      for (int j = i + 1; j < size; j++)
{
         if (arr[i] == arr[j])
 {
            printf("%d ", arr[j]);
            break;
         }
      }
   }
   printf("\n");
   return 0;
```

```
}
```

OUTPUT:

Enter the size of the array (max 100): 6

Enter elements of the array:

2 5 3 7 2 8

Duplicate elements in the array are: 2

4.Writea c prgram to find Max and Min from an array elements

PROGRAM:

```
#include <stdio.h>
#include <limits.h>
int main()
 {
   int arr[] = {10, 5, 8, 3, 12};
   int numElements = sizeof(arr) / sizeof(arr[0]);
   int max = INT_MIN;
   int min = INT_MAX;
   for (int i = 0; i < numElements; 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: 12

Minimum element in the array: 3

5.Given a number n.the task is to print the Fibonacci series and the sum of 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;
    } else
 {
        return fibonacci(n-1) + fibonacci(n-2);
    }
}
 int fibonacciSeries(int n)
    int sum = 0;
    printf("Fibonacci series: ");
    for (int i = 0; i < n; i++)
{
        int fib = fibonacci(i);
        printf("%d ", fib);
        sum += fib;
```

```
    }
    printf("\n");
    return sum;
}
int main()
{
    int n;
    printf("Enter the number of terms in Fibonacci series: ");
    scanf("%d", &n);
        int sum = fibonacciSeries(n);
    printf("Sum of Fibonacci series: %d\n", sum);
    return 0;
}
```

OUTPUT:

```
Enter the number of terms in Fibonacci series: 10
Fibonacci series: 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 arr using binary search:

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

Output:element found at location 2

Example2:arr={1,5,6,7,9,10},x=11

Output:element not found at location 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 n = sizeof(arr) / sizeof(arr[0]);
    int x;
    printf("Enter the element to search: ");
    scanf("%d", &x);
    int index = binarySearch(arr, 0, n - 1, x);
    if (index != -1) {
        printf("Element %d found at location %d\n", x, index);
    } else {
        printf("Element %d not found\n", x);
    }
    return 0;
}
```

OUTPUT:

```
Enter the element to search: 6
Element 6 found at location 2
```

Enter the element to search: 11

Element 11 not found at location 2

7.Linear search array

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int main() {
    int arr[MAX_SIZE];
    int numElements, x;
    printf("Enter number of elements in the array (max %d): ", MAX_SIZE);
    scanf("%d", &numElements);
     printf("Enter elements of the array:\n");
    for (int i = 0; i < numElements; i++) {
        scanf("%d", &arr[i]);
    }
     printf("Enter the element to search: ");
    scanf("%d", &x);
     int found = 0;
    for (int i = 0; i < numElements; i++) {
        if (arr[i] == x) {
            printf("Element %d found at location %d\n", x, i + 1);
            found = 1;
            break;
        }
    }
    if (!found) {
        printf("Element %d not found\n", x);
    }
    return 0;
```

```
}
```

OUTPUT:

Enter number of elements in the array (max 100): 5

Enter elements of the array:

10, 5, 8, 3, 12

Enter the element to search: 8

Element 8 found at location 3

8.Binary search array

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 n = sizeof(arr) / sizeof(arr[0]);
    int x;

    printf("Enter the element to search: ");
    scanf("%d", &x);
    int index = binarySearch(arr, 0, n - 1, x);
    if (index != -1) {
        printf("Element %d found at location %d\n", x, index + 1);
    } else {
        printf("Element %d not found\n", x);
    }
    return 0;
}
```

OUTPUT:

Enter the element to search: 6

Element 6 found at location 3