**Data Structures and Algorithms**

* **Data Structures**
  + **Array**
  + **Linked List**
  + **Stack**
  + **Queue**
  + **Heap**
  + **Tree**
  + **Graph**
* **Algorithms**
  + **Searching Algorithm**
  + **Sorting Algorithm**
  + **Divide and Conquer Algorithm**
  + **Recursion**

**Week - 22.12.2022**

**Data Structures**

**– *Array***

An array is a way of storing a group of things that are all the same. They are put together in a certain order, and you can easily get to any of them using a number called an "index."

*int myArray[5] = {10,20,30,40,50}; //Declaration and initialization of an array*

***Operations in Array***

**1-Traverse**

Printing all array elements

*for (int i = 0; i < 5; i++) {*

*printf("myArray[%d] = %d\n", i, myArray[i]); //print the array elements*

*}*

**2-Deletion**

**3-Insertion**

**4-Search**

**5-Update**

***Linked List***

A Linked List is a way of storing data in a specific order, similar to an array. Each data collection (called a node) is connected to the next one by a pointer (or previous one). Each node has its own data and a pointer to the next node.

***Simple Linked List***

It is also called single linked list. It is a type of linked list which can only be traversed only one direction. Each pointer of nodes points next node. The last pointer points NULL.

***Doubly Linked List***

Linked list which can be traversed in both directions as either previous or next pointers.

***Circular Linked List***

Last node’s pointer points the head of the linked list.

***Doubly Circular Linked List***

***Representation of Simple Linked List***

Typedef struct Node {

    int data;

    struct Node\* next;

}Node;

***Stack***