Cycle 1

1. Implement a menu-driven program for the following operations on the generic linear array.
    a. Insertion at Beginning
    b. Insertion at end
    c. Insertion at a specified position.
    d. Deletion from Beginning
    e. Deletion from end
    f. Deletion from a specified position.
    g. Find the index of a given element
    h. Display
2. Consider the array template implemented in Question 1 and extend the program with the various search and sorting algorithms.(quick,bubble,insertion,merge,selection,binary,linear)
3. Consider the array template implemented in Question 1 and extend the program with the advanced array operations.(left rotation,right,freq count, distinct)
4. Implement a menu-driven program for the following operations on the generic singly Linked List.
    a. Insert at Beginning
    b. Insert at End
    c. Insert at a specified Position
    d. Delete from Beginning
    e. Delete from End
    f. Delete from a specified Position
    g. Display
5. Implement a menu-driven program for the following operations on the Doubly Linked List,Circular Linked List, Circular Doubly Linked List
    a. Insert at Beginning
    b. Insert at End
    c. Insert at a specified Position
    d. Delete from Beginning
    e. Delete from End
    f. Delete from a specified Position
    g. Display
6. Implement a program for Polynomial Addition and Polynomial Multiplication using Linked List.Display the resultant polynomial.
7. implement a sorting algorithm for organizing a music playlist to organize a linked list based on different criteria such as song title, artist, duration, and genre.

Cycle 2

1. Implement a **stack using array** with the following operations :
    a. PUSH
    b. POP
    c. IS EMPTY
    d. IS FULL
    e. UNDERFLOW
    f. OVERFLOW
    g. Display
2. Implement a **stack using linked list** with the following operations :
    a. PUSH
    b. POP
    c. IS EMPTY
    d. IS FULL
    e. UNDERFLOW
    f. OVERFLOW
    g. Display
3. Implement a **queue using array** with the following operations :
    a. Insert elements to the Rear of the queue
    b. Delete elements from the Front of the queue.
    c. IS EMPTY
    d. IS FULL
    e. UNDERFLOW
    f. OVERFLOW
    g. Display
4. Implement a **queue using linked list** with the following operations :
    a. Insert elements to the Rear of the queue
    b. Delete elements from the Front of the queue.
    c. IS EMPTY
    d. IS FULL
    e. UNDERFLOW
    f. OVERFLOW
    g. Display
5. Implement a **Double-Ended Queue (DEQUEUE) using array** with the following operations :
    a. Insert elements to the Front of the queue.
    b. Insert elements to the Rear of the queue
    c. Delete elements from the Front of the queue.
    d. Delete elements from the Rear of the queue.
    e. Display the queue after each operation.
6. Implement a **Double-Ended Queue (DEQUEUE) using Linked list** with the following operations:

   a. Insert elements to the Front of the queue.
   b. Insert elements to the Rear of the queue
   c. Delete elements from the Front of the queue.
   d. Delete elements from the Rear of the queue.
   e. Display the queue after each operation.

7. Implement a two - way stack with its operations.
8. Convert an infix expression to a postfix expression as well as prefix expression using stack.
9. Evaluation of Postfix expression using stack
10. Implement a program to check the Balanced Bracket expression using Stack.
11. Implement a program to implement a recursion using stack