1 | Experiential Learning Component

1.1 | UNIT II : Experiential Learning - 4

[Level-1: 4Q, Level-2: 1Q, Level-3: 2Q]

1. **LEVEL - 1 : Create Circular Linked List:** Program a function to initialize a circular linked list with 'n' nodes, inserting values from 1 to 'n'.
   - Hint: The last node's next pointer should refer to the head of the list.
2. **LEVEL - 1 : Delete by Value in Circular List:** Code a function to delete all nodes with a specific value in a circular linked list.
   - Hint: Address the special case where the head node contains the value.
3. **LEVEL - 2 : Sorted Insert in Circular List:** Devise a function to insert a node into a sorted circular linked list, preserving the order.
   - Hint: Determine the correct insertion point by comparing values.
4. **LEVEL - 3 : Circular List Tail Attachment:** Code a function to append a separate linked list at the end of a circular linked list.
   - Hint: Connect the two lists and ensure the tail of the new list points to the head of the circular list.
5. **LEVEL - 1 : Doubly Linked List Insertion:** Write a function to insert a new node before a given node in a doubly linked list.
   - Hint: Modify the previous and next pointers of the relevant nodes.
6. **LEVEL - 3 : Doubly Linked List Deletion:** Program a function to delete nodes with even values from a doubly linked list.
   - Hint: Traverse the list and free the nodes carefully to avoid memory leaks.
7. **LEVEL - 1 : Implement Stacks Using Linked Lists (both Singly LL and Doubly LL)** Create a program to implement Stacks .
   - Hint: Re-use the functions used to implement primitive operations on Linked Lists (Create a wrapper function which will invoke the previously created function).