Exercise: 05	Linked List	31 – Jul - 2026

Observation (5 Marks)

1. Why does the last node point to the first node in a circular linked list?
2. Write the time complexity of insertion at the beginning of a circular linked list.
3. How is overflow avoided in a linked-list implementation of a queue?
4. Write a C program to implement enqueue and dequeue operations using a circular linked list.
5. Explain the following
    a. What are the two links present in a doubly linked list node?
    b. What is the advantage of a doubly linked list over a singly linked list?
    c. How can backward traversal be performed in a doubly linked list?
    d. What is stored in the previous pointer of the first node?
    e. State one application of a doubly linked list.6. How can you detect a cycle in a linked list?

Execution (15 Marks)

1. Design a C program to create a circular linked list representing a music playlist. Each node stores a song ID. Display all songs in the playlist starting from the first song and continuing until all songs are displayed once.

   Sample Input:

   Songs: 101, 102, 103, 104

   Expected Output:

   Playlist: 101 → 102 → 103 → 104 → Back to 1012.

2. Develop a Menu-Driven Program for Circular Queue Operations

   For Call Center Customer Support using a circular queue implemented with a linked list to manage customer service requests.

A customer support center receives service requests continuously. Requests are handled in the order they arrive. New requests can be added, processed, searched, or displayed.

a. Add a customer request (Enqueue)
b. Process a request (Dequeue)
c. Search for a request ID
d. Display all pending requests
e. Exit

Sample Input:

Requests: 201, 202, 203

Process Request: 201

Search Request: 202

Expected Output:

Pending Requests: 202 → 203 → Back to 202

Request 202 Found.

3. A photo gallery application allows users to move to the next image or return to the previous image. Implement a doubly linked list to store photo IDs. Display

a. Photos in forward direction.
b. Photos in reverse direction.

Sample Input:
Photo IDs: P1, P2, P3, P4

Expected Output:

Forward Traversal:
P1 ⇄ P2 ⇄ P3 ⇄ P4

Backward Traversal:
P4 ⇄ P3 ⇄ P2 ⇄ P1