Riphah International University
I-14 Main Campus
Faculty of Computing

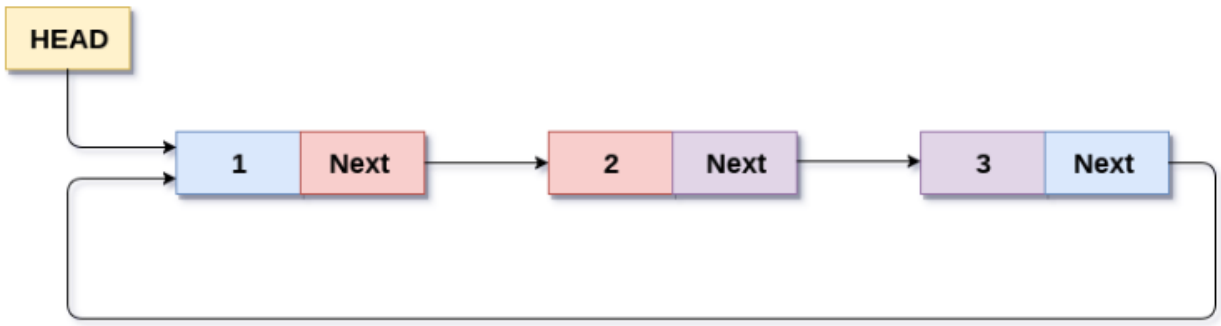
Class:	Fall-2024	**Subject:**	Data Structures & Algorithms
Course Code:	CS 2124	**Lab Instructor:**	Zeeshan Ali

Learning Objective:

- Circular linked list
- Circular linked list operations
- Implementation of Circular linked list in C++
- Operations on Circular linked list
- Insert
- printList
- Revision of link list deletion
- Lab Task

Circular linked list:

Circular linked list is a sequence of elements in which **all** nodes are connected to form a **circle**. In a circular linked list, the **first** node and the **last** node are connected to each other which forms a circle. There is **no NULL** at the end.



In a circular singly linked list, the **last node** of the list **contains** a **pointer** to the **first node** of the list. We traverse the circular singly linked list until we reach the same node where we started. The circular singly linked list has no beginning or end. No null value is present in the next part of any of the nodes.

Circular linked list operations:

- Insertion
- Display

Implementation of Double linked list in C++

CircularLinkedListV1.cpp

```cpp
#include <iostream>
using namespace std;
class Node {
    public:
    int data;
    Node* next;
};

class CircularLinkedList
{
    private:
    Node* head;

    public:
    CircularLinkedList() {
        head = NULL;
    }

```

Insert

```
19  void insert(int data)
20      {
21          Node* newNode = new Node;
22          newNode->data = data;
23          newNode->next = NULL;
24
25          if (head == NULL) {
26              head = newNode;
27              head->next = head;
28          }
29          else
30          {
31              Node* temp = head;
32              while (temp->next != head)
33              {
34                  temp = temp->next;
35              }
36              temp->next = newNode;
37              newNode->next = head;
38          }
39      }
40
```

printList

```
41  void printList() {
42          if (head == NULL) {
43              cout << "List is empty." <<endl;
44              return;
45          }
46
47          Node* current = head;
48          do {
49              cout << current->data << "\t";
50              current = current->next;
51          } while (current != head);
52
53      }
54  };
```

```
55
56  int main() {
57      CircularLinkedList clist;
58      clist.insert(5);
59      clist.insert(7);
60      clist.insert(9);
61      clist.insert(11);
62      cout << "Circular Linked List: ";
63      clist.printList();
64      return 0;
65  }
66
```