TASK 3

Project Report

Title:

Implementation of Stack and Queue Using Linked Lists in C++

Objective:

To design and implement two fundamental data structures — **Stack** and **Queue** — using **linked lists** in C++ and provide basic operations like push, pop, enqueue, dequeue, and display.

Tools Used:

- **Programming Language:** C++
- **Compiler:** Any C++ compiler (e.g., GCC, Code::Blocks, Visual Studio)

Introduction:

Stacks and Queues are core concepts in Data Structures.

- A **Stack** follows **Last-In-First-Out (LIFO)**, where the last inserted element is the first to be removed.
- A **Queue** follows **First-In-First-Out (FIFO)**, where the first inserted element is the first to be removed.

In this project, both structures are implemented using **linked lists**, allowing dynamic memory allocation and avoiding limitations like fixed size.

Methodology:

1. Stack Implementation:

- A Node structure is used with two members: data and a pointer next.
- **Push operation** inserts a new node at the top of the stack.
- **Pop operation** removes the node from the top.
- **Display operation** traverses the stack and prints all elements.

2. Queue Implementation:

- The same Node structure is used.
- **Enqueue operation** inserts a new node at the rear of the queue.
- **Dequeue operation** removes a node from the front.
- **Display operation** traverses the queue and prints all elements.

3. User Interaction:

- A **menu-driven** approach is used, allowing users to choose operations repeatedly until they choose to exit.

Program Flow Diagram:

mathematica

CopyEdit

Start

↓

Display Menu

↓

Get User Choice

↓

Perform Operation:

- Stack: Push / Pop / Display
- Queue: Enqueue / Dequeue / Display

↓

Repeat until user selects Exit

↓

End

Code Structure:

- **Node structure:** Contains integer data and a next pointer.

- **Stack class:**
  - push(int value)
  - pop()
  - display()
- **Queue class:**
  - enqueue(int value)
  - dequeue()
  - display()
- **Main function:** Handles user input and directs to the appropriate operation.