HOSPITAL EMERGENCY QUEUE SYSTEM

Name: ISHTIAQ AHMED

Roll No: 14

PROBLEM STATEMENT

The hospital's Emergency Room needs a system to manage patients by priority. Using a **Doubly Linked List**, each patient (node) can be added or removed efficiently from the beginning, end, or a specific position.

This ensures critical patients are treated first and the ER queue updates dynamically.

The system must support the following operations:

- Insert a patient at the beginning (critical patient arrives)
- Insert a patient at the end (normal walk-in patient)
- Insert a patient at a specific position (as decided by the nurse)
- Delete a patient from the beginning (patient treated and leaves)

PROPOSED SOLUTION

The system uses a **Doubly Linked List** to manage ER patients efficiently. Each node stores a **patient ID** and links to both previous and next patients. The list is controlled by two pointers: **head** (first patient) and **tail** (last patient).

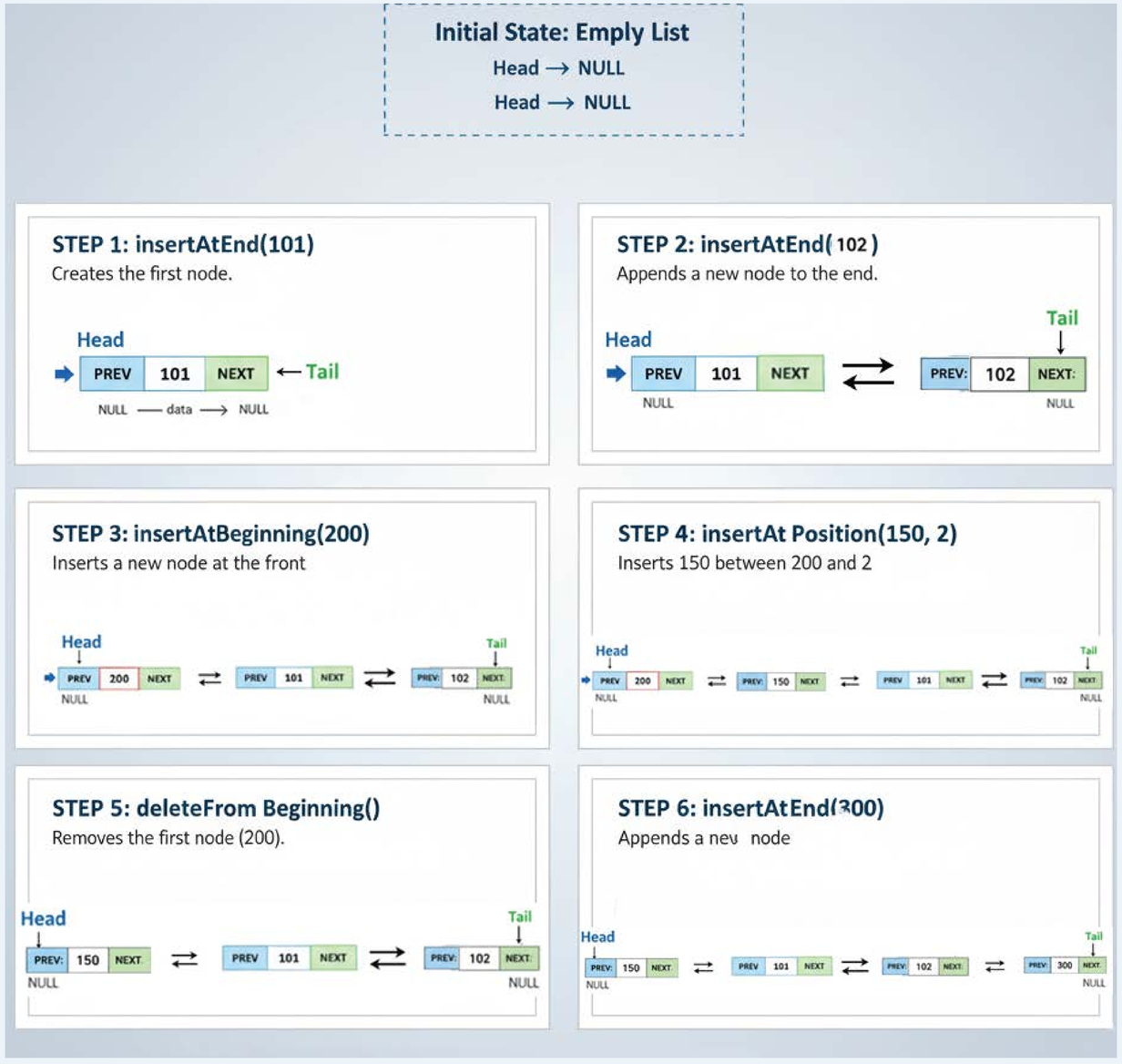
Operations Supported:

- **insert_at_beginning:** Adds a critical patient at the start.
- **insert_at_end:** Adds a normal patient at the end.
- **insert_at_position:** Inserts a patient at a chosen priority position.
- **delete_from_beginning:** Removes the first patient after treatment.

Graphical View of ER Queue Operations

The following diagram illustrates the step-by-step changes in the Emergency Room queue after performing these operations:

- insertAtEnd(101)
- insertAtEnd(102)
- insertAtBeginning(200) // critical patient
- insertAtPosition(150, 2)
- deleteFromBeginning()
- insertAtEnd(300)



WORKING METHODOLOGY

The Emergency Room Queue System is implemented using a **Doubly Linked List** to efficiently manage patient priority. The methodology involves the following steps:

1. **Initialization:** Create a doubly linked list where each node contains a **patient ID** and pointers to the **previous** and **next** nodes.
2. **Patient Insertion:**
   - **At the beginning:** For critical patients who need immediate attention.
   - **At the end:** For normal walk-in patients who can wait.
   - **At a specific position:** As per nurse's priority decision.
- **Patient Deletion:**
  - Remove the **first patient** from the list when treatment is completed.
- **Pointer Management:**
  - After each operation, update the **head**, **tail**, **next**, and **previous** pointers to maintain list integrity.
- **Queue Maintenance:**
  - Keep track of the **total number of patients**.
  - Handle edge cases such as empty lists or single-node lists.

CONCLUSION

The Emergency Room Queue System using a **Doubly Linked List** efficiently manages patients based on priority. It allows quick insertion and deletion of patients from any position, ensuring critical cases are treated first. This approach keeps the ER list dynamic, organized, and easy to update in real time.