Detailed Dry Run of Queue using LinkedList (Code-Mapped)

Step 1: Create Empty Queue

- Constructor called: front=null, rear=null
- Queue is empty.

Diagram:
front=null, rear=null

Step 2: insertAtRear(10)

- Code check: if (rear==null) → true
- Action: rear=node; front=node;
- Thought: Queue empty tha, ab ek hi node hai.

Diagram:
front → [10] ← rear

Step 3: insertAtRear(20)

- Code check: if (rear==null) → false
- Action: rear.next=node; rear=node;
- Thought: 20 ko rear ke baad joda, rear aage badh gaya.

Diagram:
front → [10] → [20] ← rear

Step 4: insertAtRear(30)

- Code check: if (rear==null) → false
- Action: rear.next=node; rear=node;
- Thought: 30 rear me add hua, rear shift ho gaya.

Diagram:
front → [10] → [20] → [30] ← rear

Step 5: getFront() & getRear()

- getFront: return front.item → 10
- getRear: return rear.item → 30
- Thought: FIFO me front hamesha pehle element ko point karega, rear last ko.

Step 6: delete() once

- Code check: if (isEmptyQueue()) → false
- front==rear? → false (multiple nodes)
- Action: front=front.next
- Thought: 10 remove, ab front 20 ko point karega.

Diagram:

front → [20] → [30] ← rear

Step 7: delete() second time

- Code check: queue not empty
- front==rear? → false
- Action: front=front.next
- Thought: 20 remove, ab front 30 ko point karega.

Diagram:
front → [30] ← rear

Step 8: delete() third time

- Code check: queue not empty
- front==rear? → true (single node)
- Action: front=null; rear=null;
- Thought: Queue empty ho gaya.

Diagram:
front=null, rear=null

Step 9: delete() on empty queue

- Code check: isEmptyQueue() → true
- Action: throw Queue Underflow Exception
- Thought: Empty queue se delete possible nahi.