SNS COLLEGE OF TECHNOLOGY

Coimbatore-35

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

COURSE NAME : COURSE NAME : 23ITT101&PROGRAMMING IN C AND DATA STRUCTURES

I YEAR/ II SEMESTER

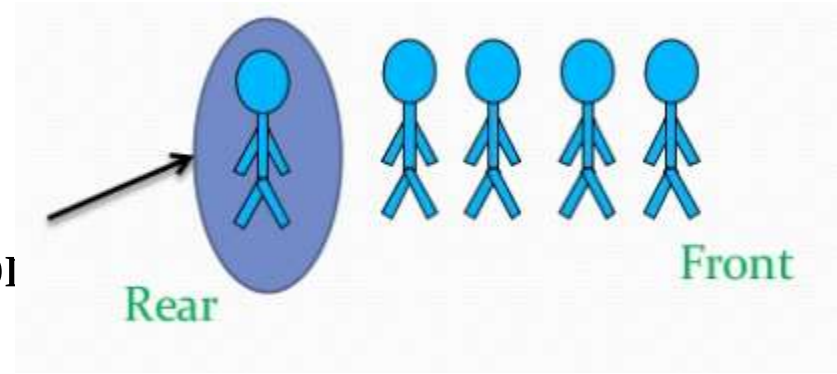
UNIT – IV STACK AND QUEUE

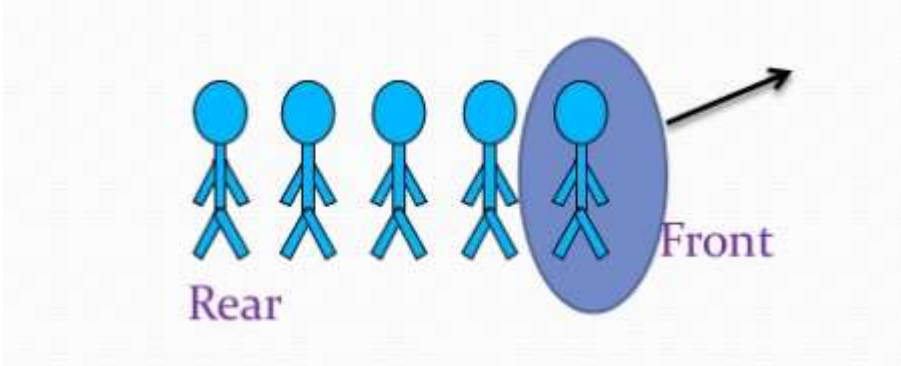
Topic: Queue ADT

Guess

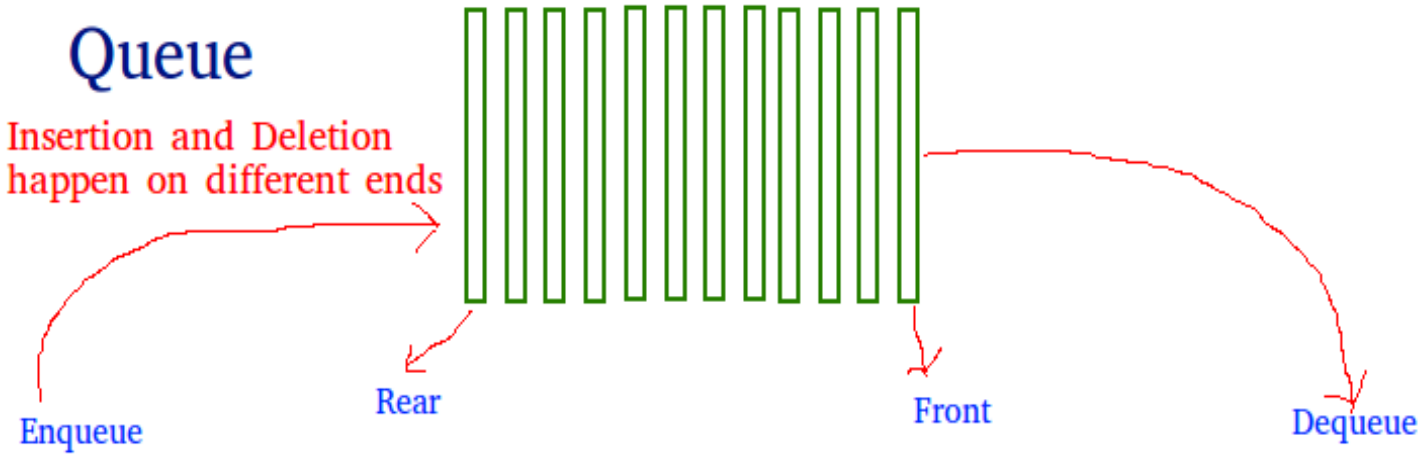
Queue -Introduction

- It is a Linear Data Structure
- Elements are in Sequential manner
- Follows FIFO-First in First out Mechanism
- Example :Standing on a line for Ticket reservation
- Two Operations
  - ENQUEUE-Inserting an element into the Queue
  - DEQUEUE-Deleting an element from the Queue
- Two Conditions
  - Overflow-Insertion into Queue which is full
  - Underflow-Deletion from Empty Queue



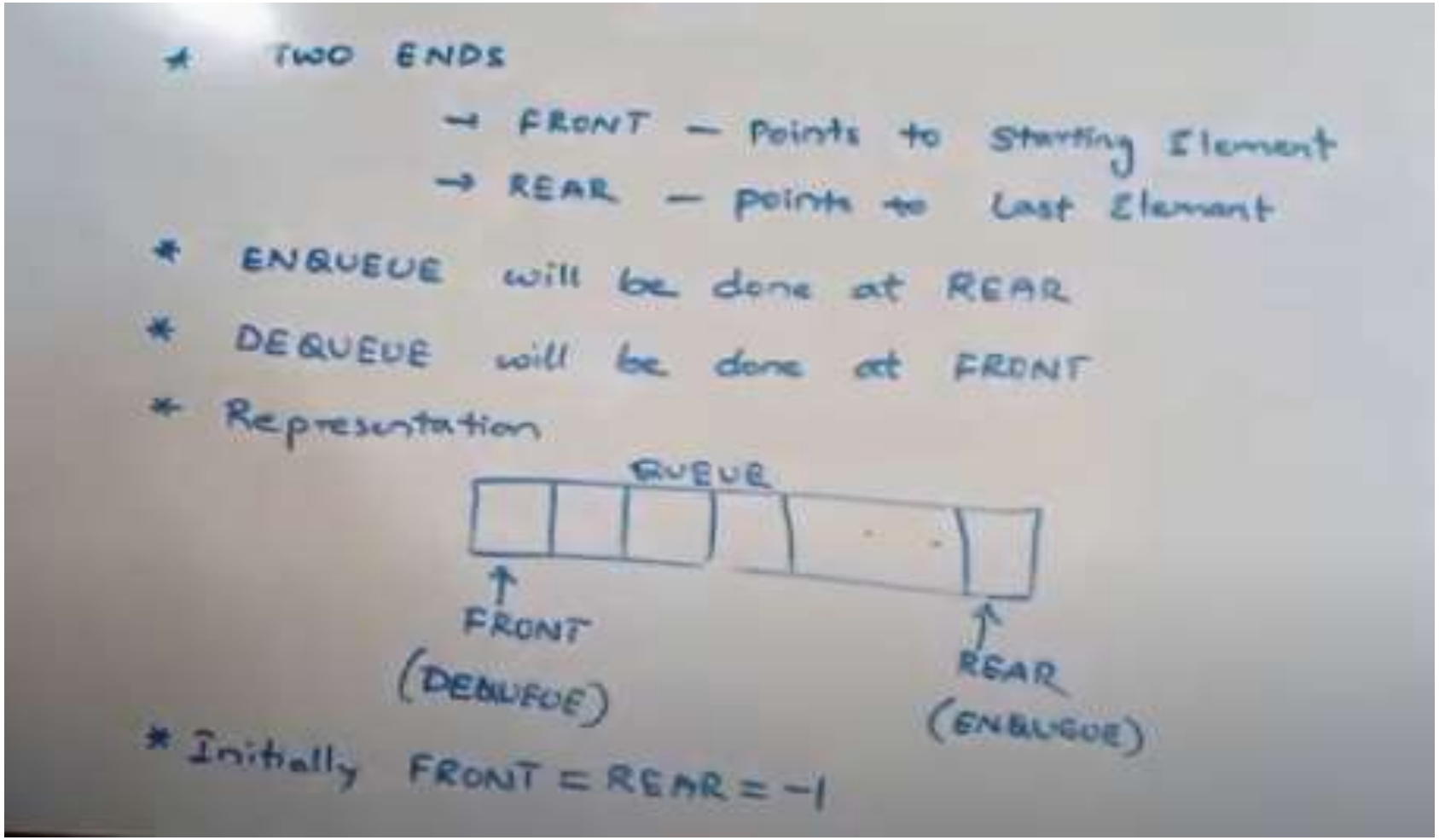


Introduction – Queue

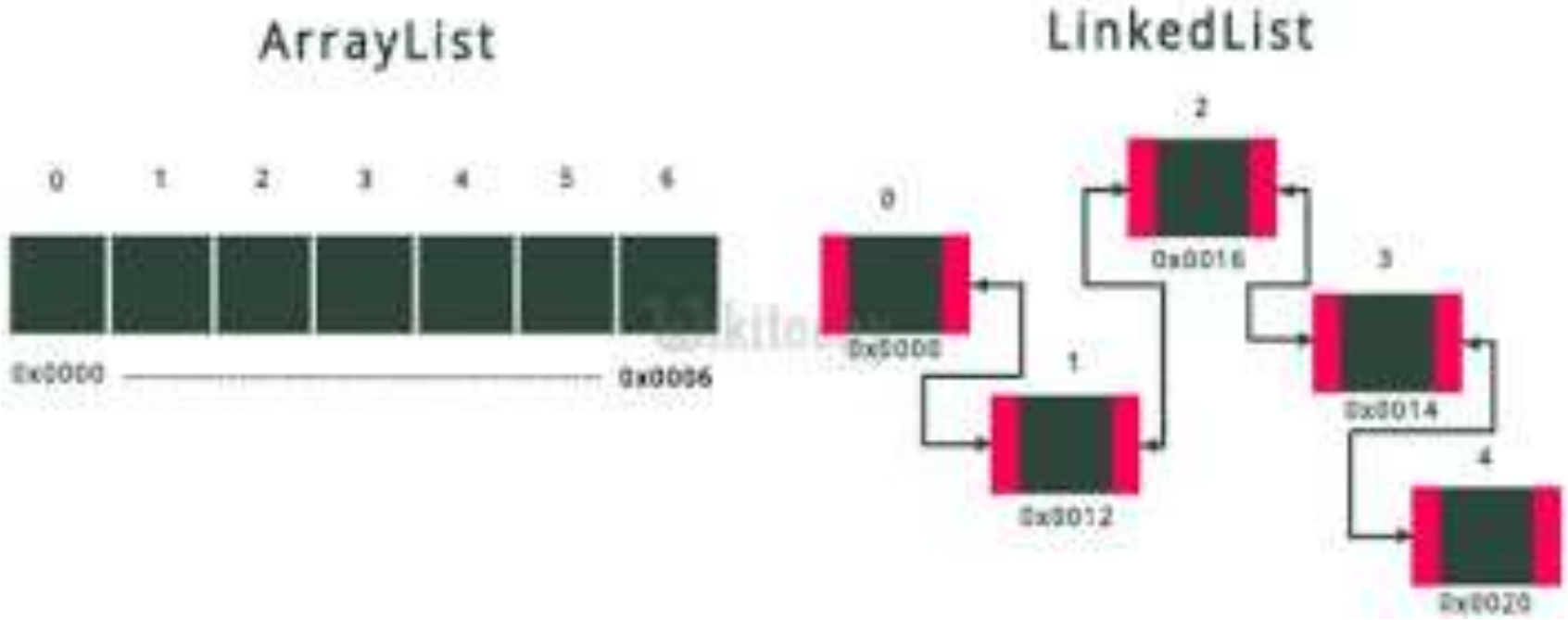


First in, first out

Queue -Representation



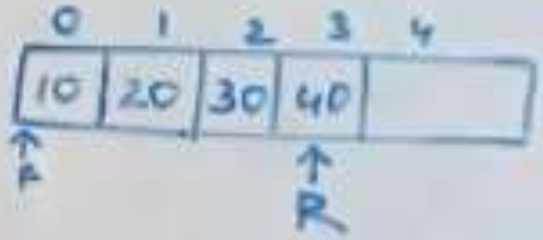
Queue -Way of Implementation

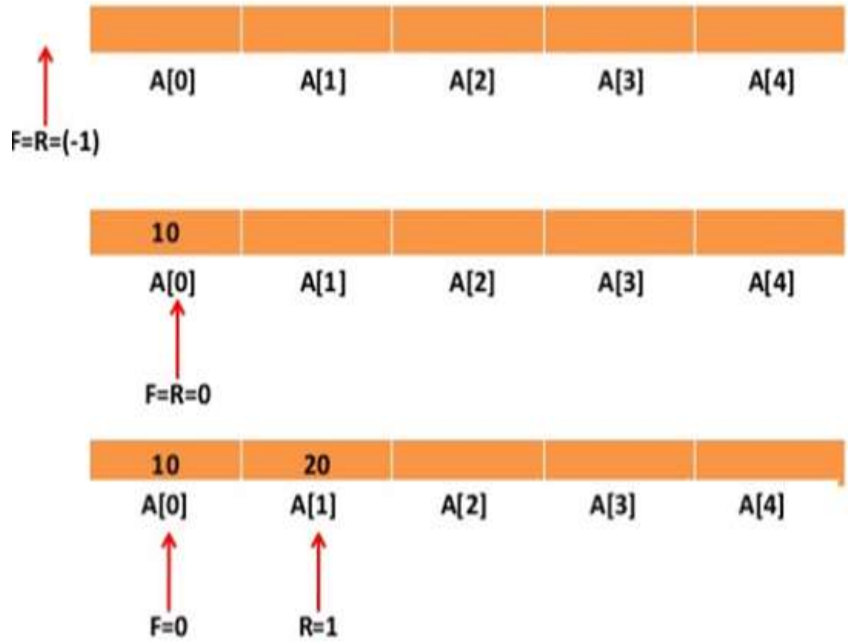


Implementation Queue Using Array-Insertion

```
SIZE = 10                    | QUEUE [ ] ;
FRONT = REAR = -1

ENQUEUE ( )
{
    if ( REAR == SIZE -1 )
        Printf ( " OVERFLOW " );
    else
    {
        scanf ( " %d ", &ele ); — 10, 20, 30, 40
        if ( FRONT == -1 )
        {
            FRONT = 0;
        }
        REAR++;
        QUEUE [REAR] = ele;
    }
}
```

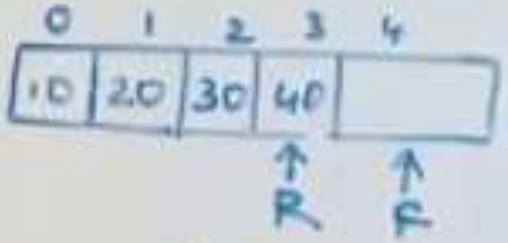




Implementation Queue Using Array-Deletion

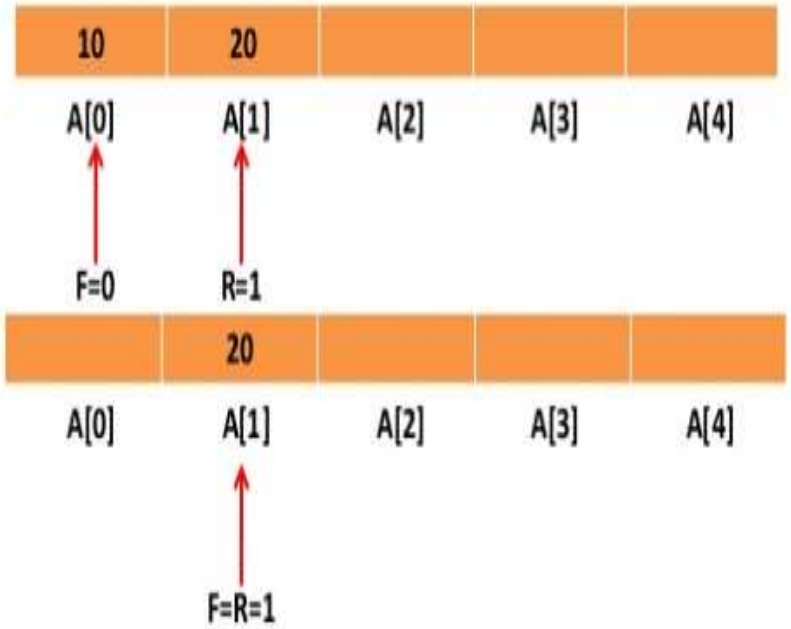
```
SIZE = 10                    | QUEUE [ ];
FRONT = REAR = -1

DEQUEUE ( )
{
    if (REAR == -1 || FRONT > REAR)
        Printf ("UNDERFLOW");
    else
    {
        ele = QUEUE[FRONT]
        Printf ("%d is deleted", ele);
        FRONT++;
    }
```

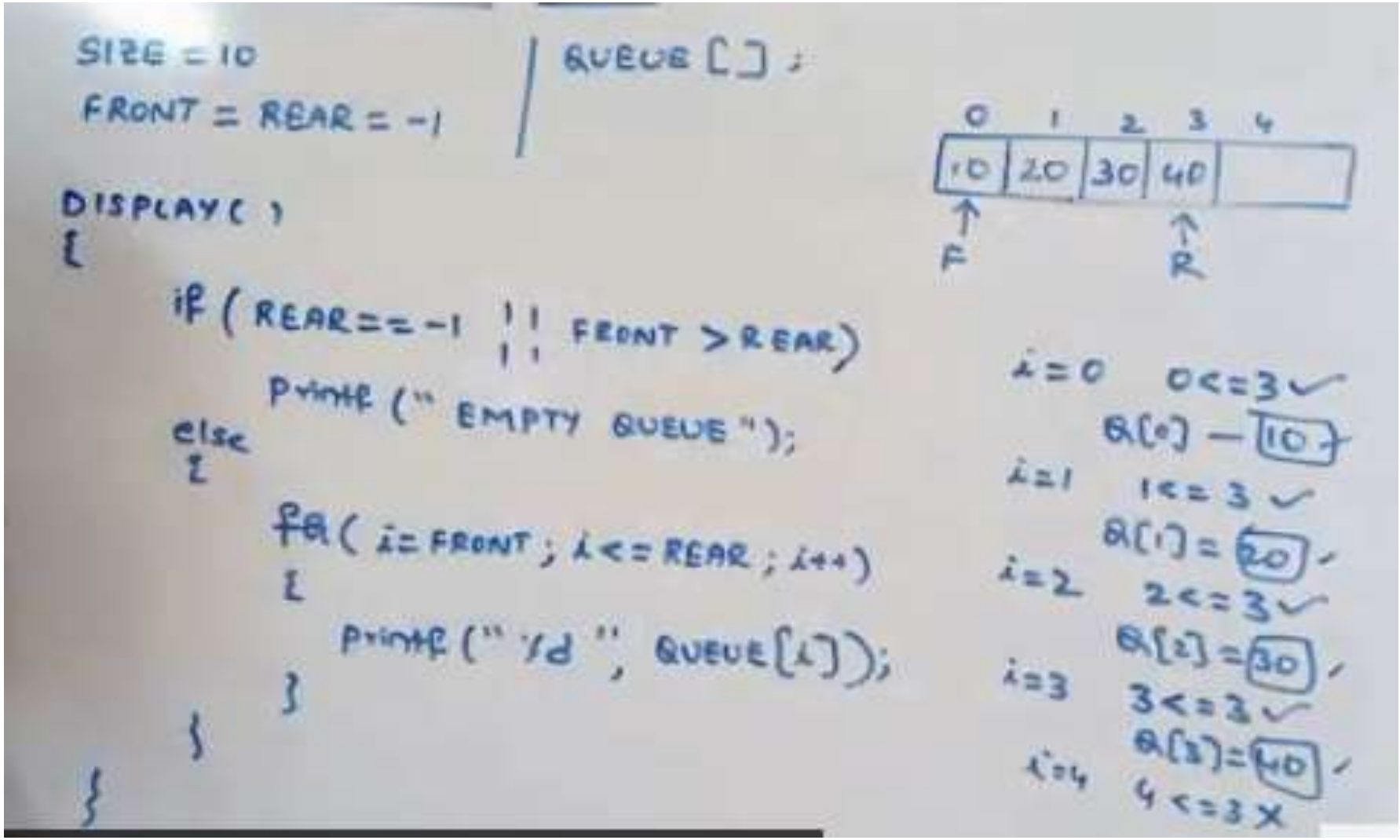


4 > 3

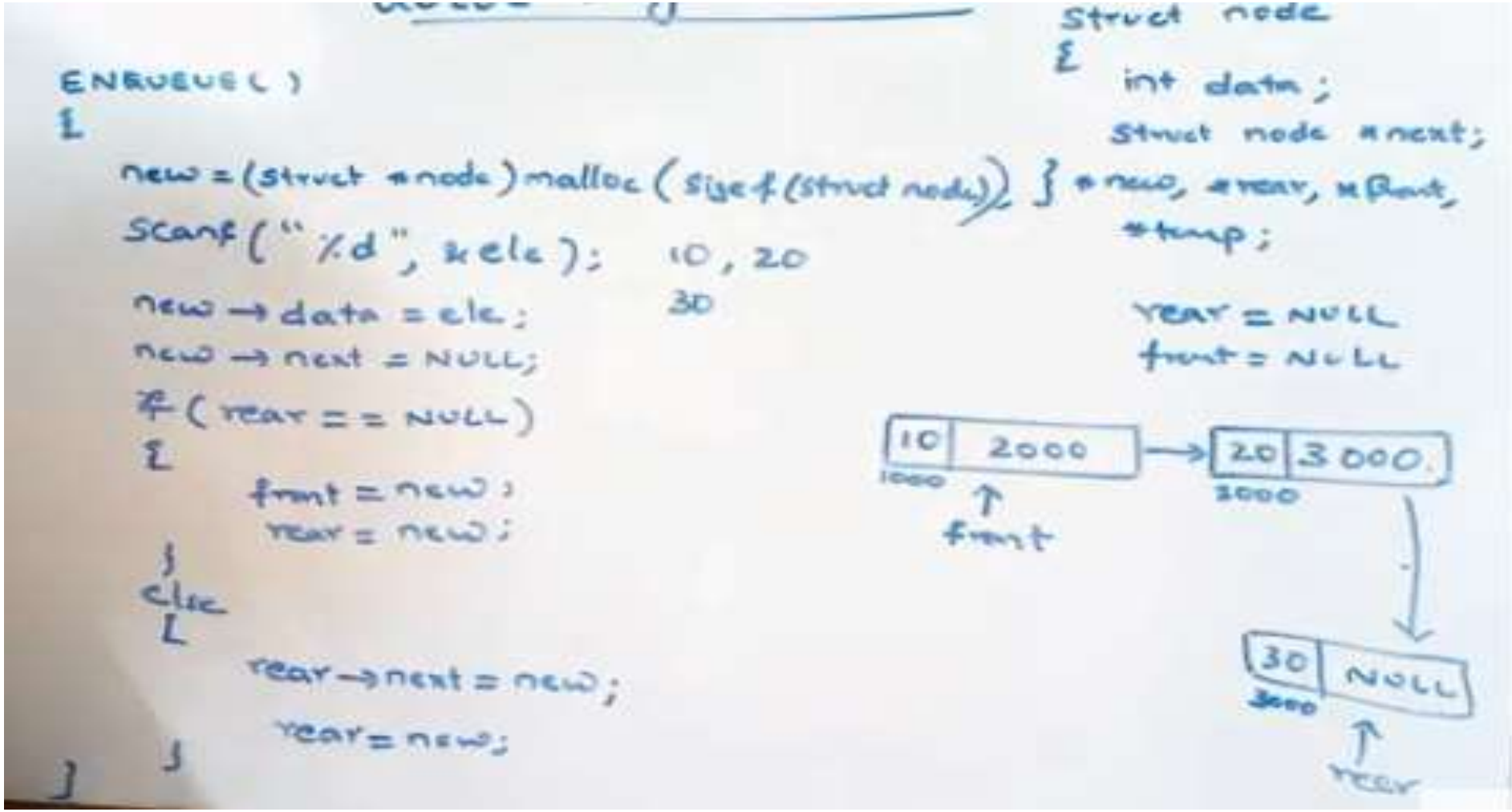
10 x
20 x
30 x
40 x



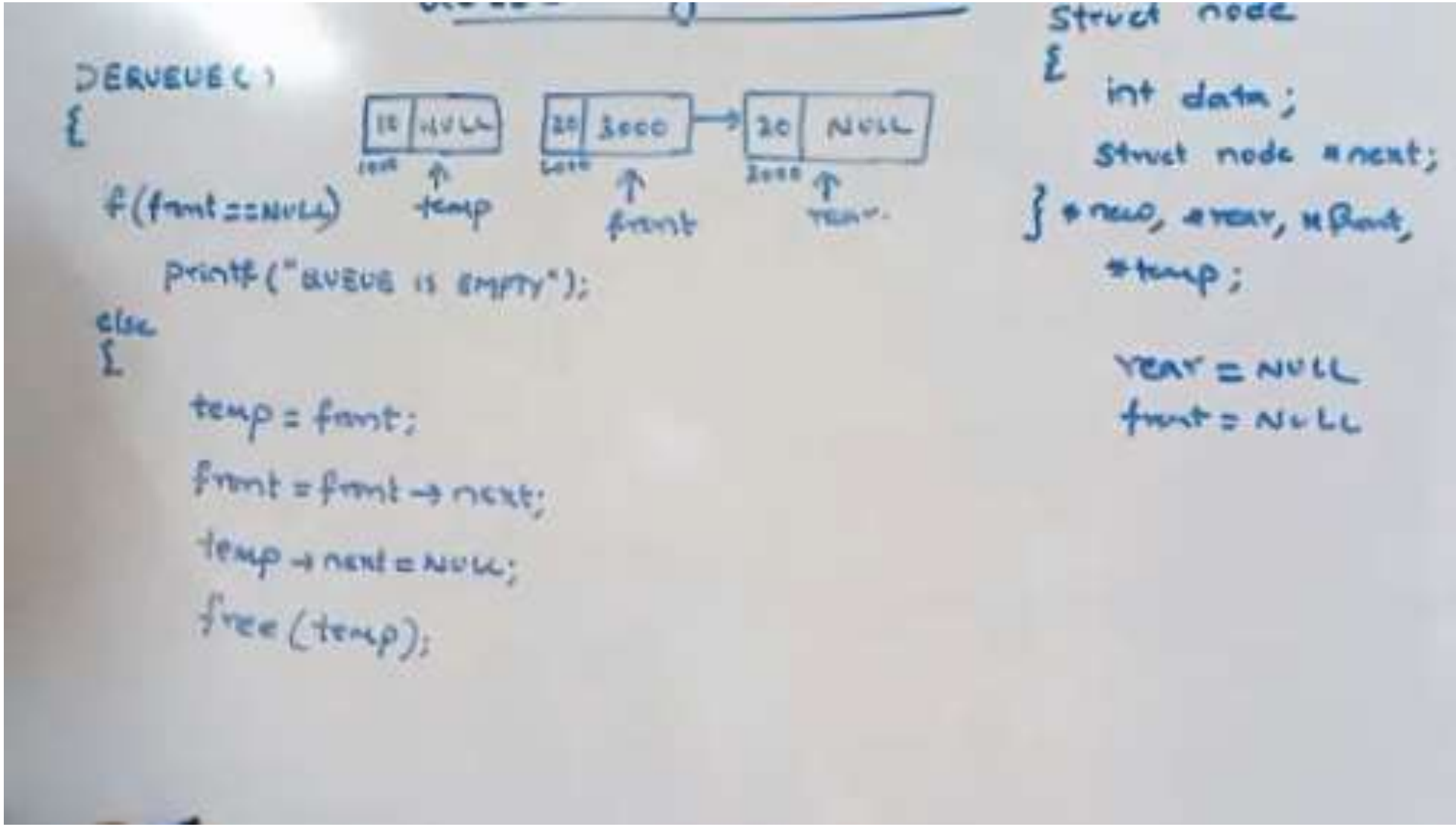
Implementation Queue Using Array-Display



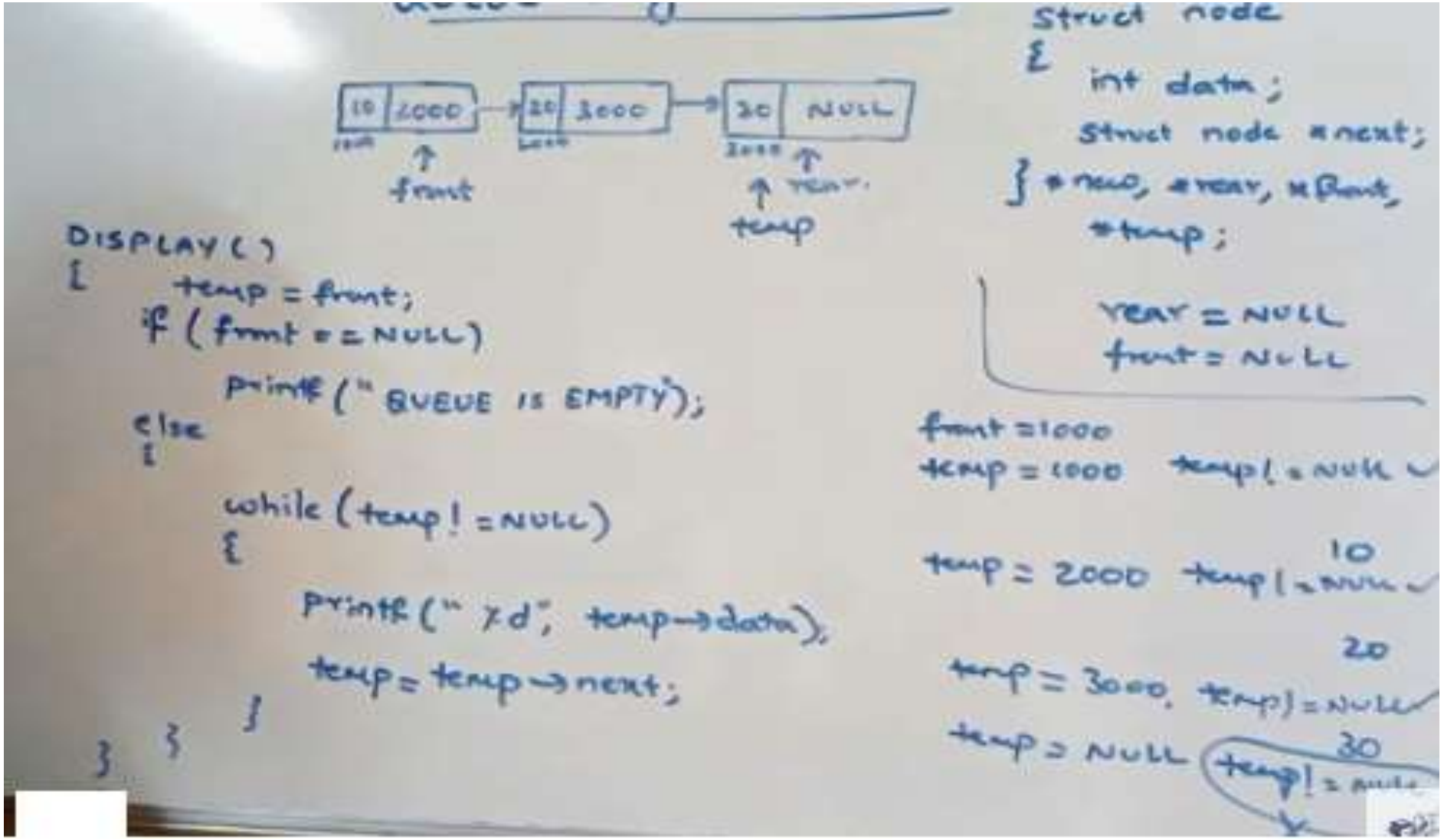
Implementation Queue Using Linked List- Insertion



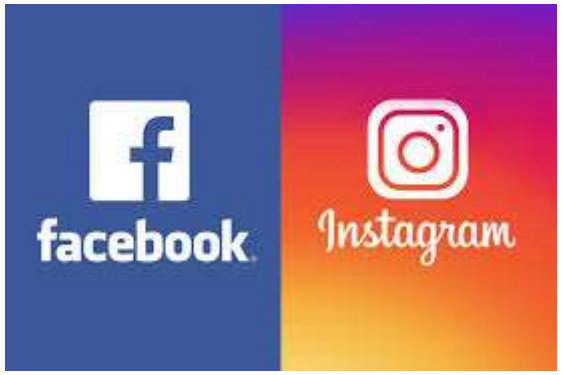
Implementation Queue Using Linked List- Deletion



Implementation Queue Using Linked List- Display

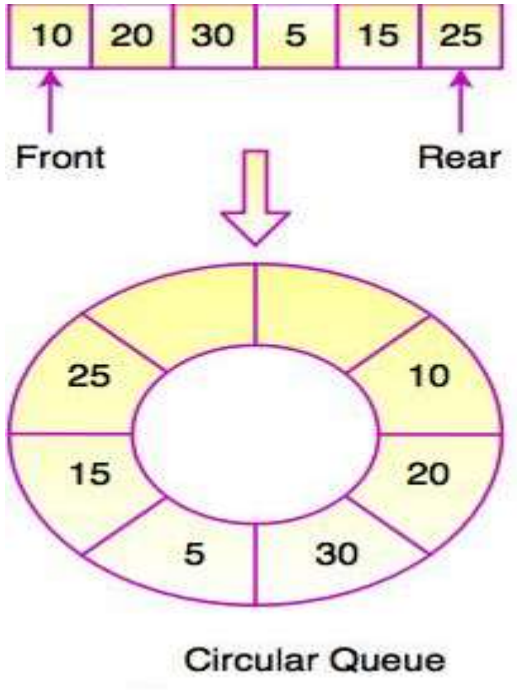


Applications



Circular Queue Implementation

- In a circular queue, all nodes are treated as circular. Last node is connected back to the first node.
- Circular queue is also called as **Ring Buffer.**
- It is an abstract data type.
- Circular queue contains a collection of data which allows insertion of data at the end of the queue and deletion of data at the beginning of the queue.
- One of the benefits of the **circular queue** is that **we** can make use of the spaces in front of the **queue**. In a normal **queue**, once the **queue** becomes full, **we** cannot insert the next element even if there is a space in front of the **queue**. But using the **circular queue**, **we** can use the space to store new values.

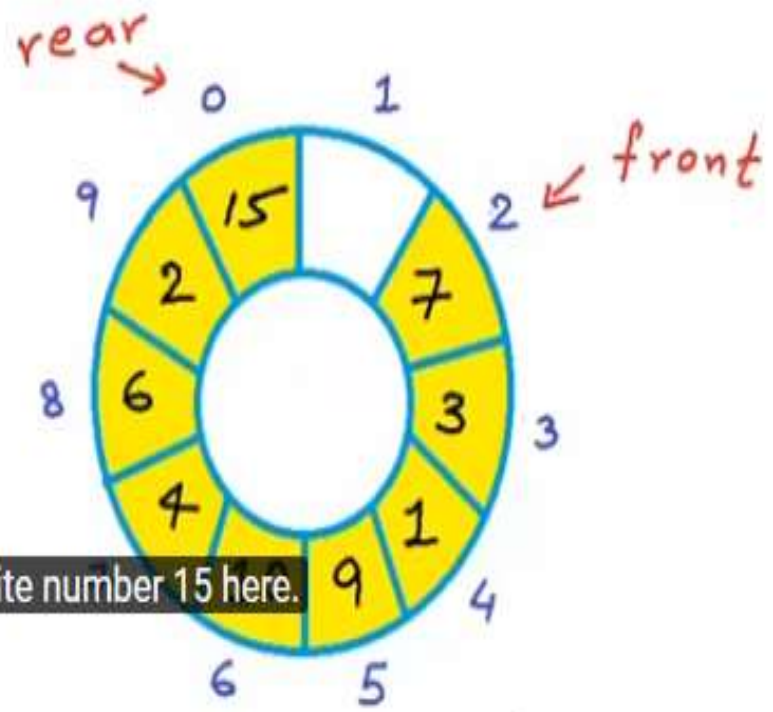


Circular Queue

Circular Queue Implementation

```
Enqueue(x)
{ if (rear+1)%N == front
      return
  else if IsEmpty()
  {
    front ← rear ← 0
  }
  else
  {
    rear ← (rear+1) % N
  }
  A[rear] ← x
```

(9+1) % 10 = 0



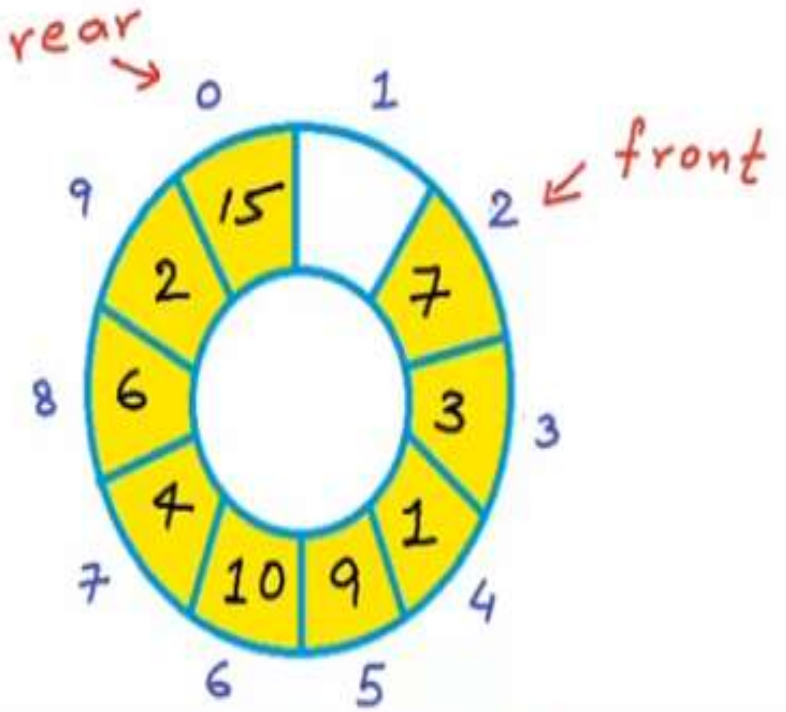
is 0. I'll write number 15 here.

Enqueue(15)

Circular Queue Implementation

```
Dequeue()
{
  if IsEmpty()
      return
  else if front == rear
      front ← rear ← -1
  else
      front ← (front +1) % N
}
```

(2+1) %



Enqueue(15)

Dequeue()

References

1. M. A. Weiss, “Data Structures and Algorithm Analysis in C”, Pearson Education, 2nd Edition, 2002.
2. A. V. Aho, J. E. Hopcroft and J. D. Ullman, “Data Structures and Algorithms”, Pearson Education, 2nd Edition, 2007
3. Ashok Kamthane, " Data Structures Using C ", Pearson Education, 2nd Edition, 2012.
4. Sahni Horowitz, “Fundamentals of Data Structures in C”Universities Press; Second edition 2008

Thank You