DSA SERIES-SESSION 6

QUEUES

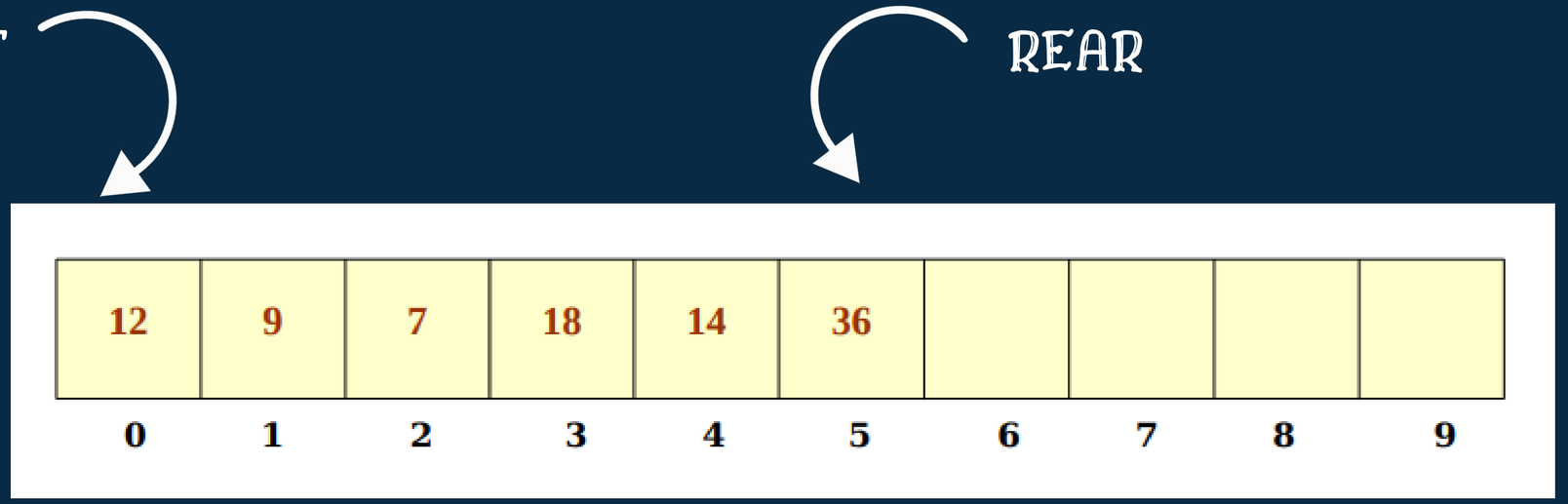
SHREYA MOHANTY

ENQUEUE

DEQUEUE

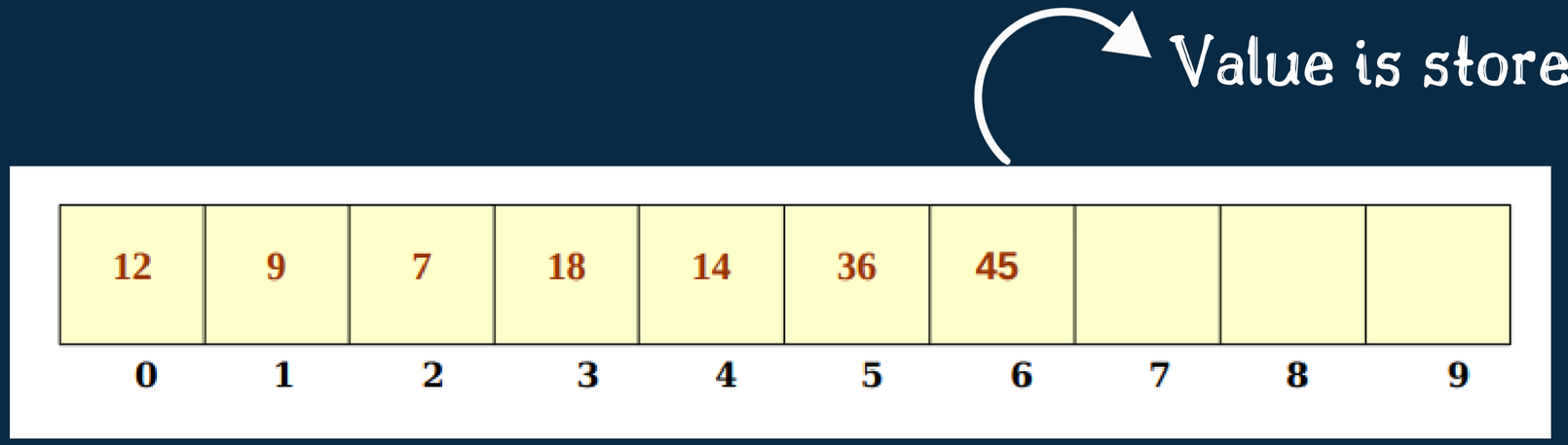
FIFO

First In First Out



Every queue has front and rear variables that point to the position from where deletions and insertions can be done

INSERTION/ENQUEUE



Before inserting an element in the queue we must check for overflow conditions. An overflow occurs when we try to insert an element into a queue that is already full, i.e. when rear = MAX – 1, where MAX specifies the maximum number of elements that the queue can hold.

ALGORITHM FOR INSERTION IN QUEUE

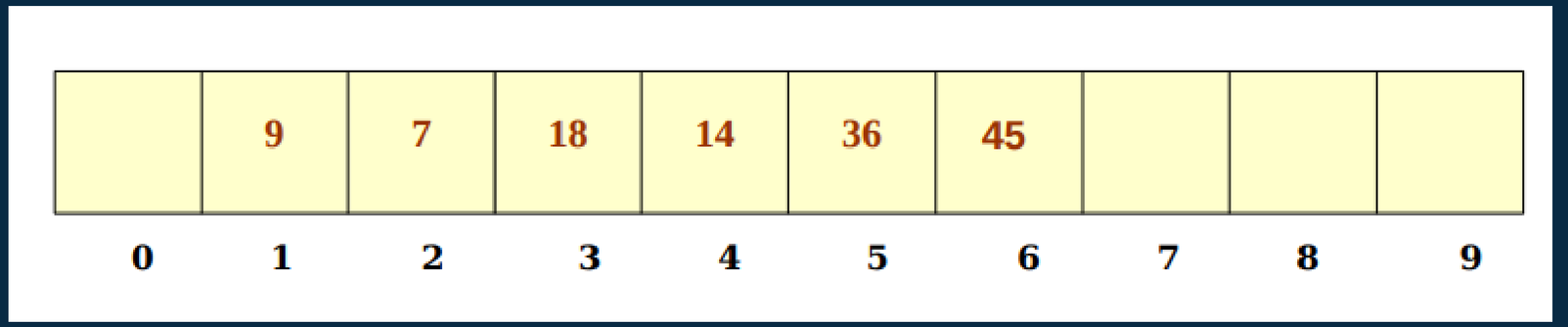
```
Step 1: IF REAR = MAX-1
            Write OVERFLOW
            Goto step 4
        [END OF IF]
Step 2: IF FRONT = -1 and REAR = -1
            SET FRONT = REAR = 0
        ELSE
            SET REAR = REAR + 1
        [END OF IF]
Step 3: SET QUEUE[REAR] = NUM
Step 4: EXIT
```

Conditions are checked

Operation is carried out

DELETION/DEQUEUE

FRONT+ 1



Value is deleted from here

Before deleting an element from the queue, we must check for underflow condition.

Index	0	1	2	3	4
Value					

- front = -1
- rear = -1

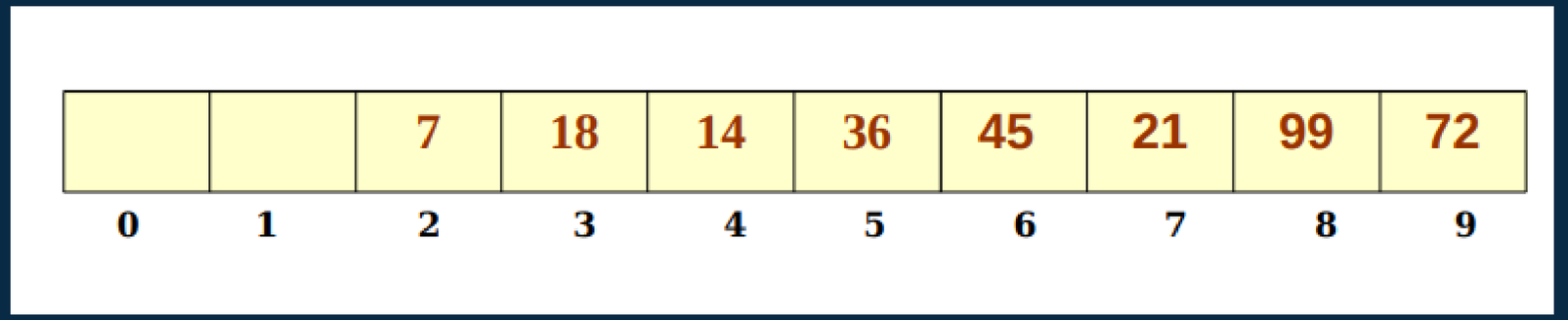
Index	0	1	2	3	4
Value					

- front = -1
- rear = -1

ALGORITHM FOR DELETION IN QUEUE

```
Step 1: IF FRONT = -1 OR FRONT > REAR
            Write UNDERFLOW
        ELSE
            SET VAL = QUEUE[FRONT]
            SET FRONT = FRONT + 1
        [END OF IF]
Step 2: EXIT
```

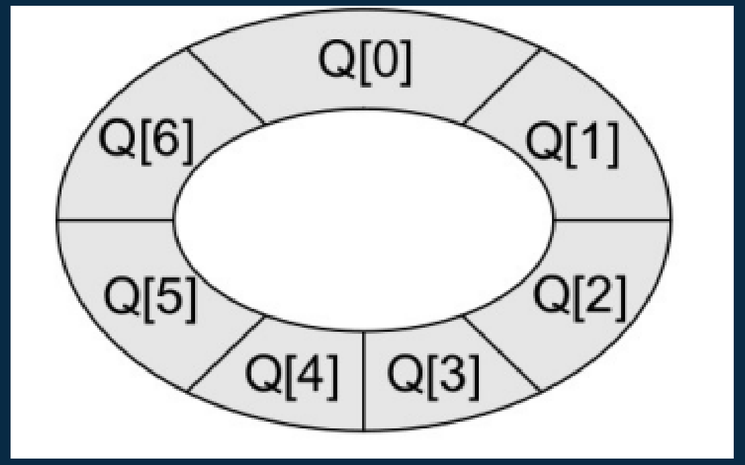
CIRCULAR QUEUES



front = 2 and rear = 9

This is the major drawback of a linear queue. Even if space is available, no insertions can be done once rear is equal to MAX – 1.

Overflow condition is achieved



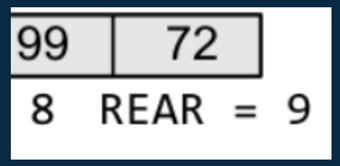
0

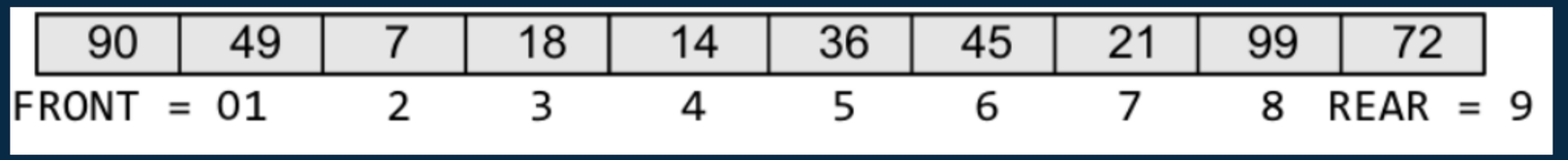
Max – 1.

In a circular queue, the first index comes right after the last index

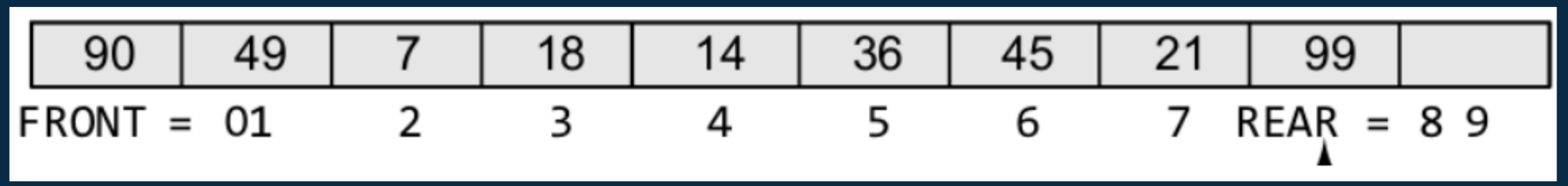
A circular queue is full, only when front = 0 and rear = Max – 1.

INSERTION IN CIRCULAR QUEUE

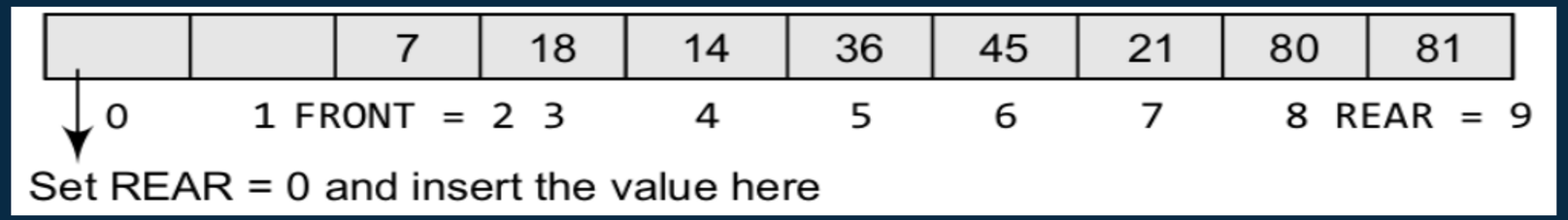


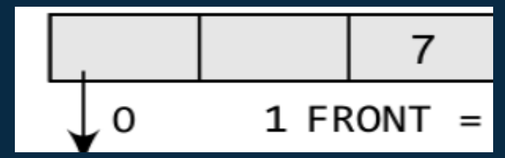


If front=0 and rear= MAX – 1 or rear = front-1, then the circular queue is full.



If rear != MAX – 1, then the rear will be incremented and value will be inserted



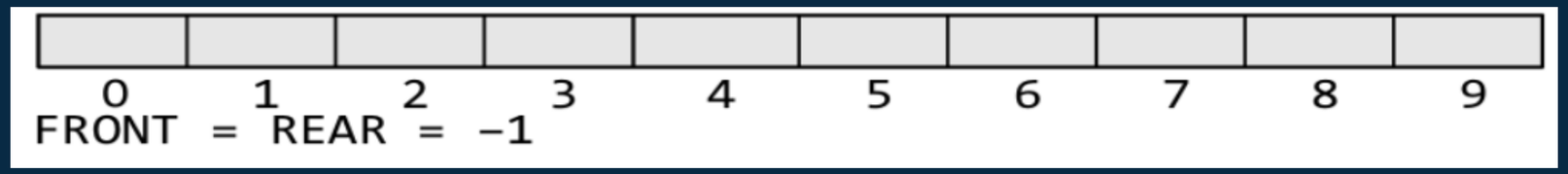


Element will be inserted at the front by setting rear=0

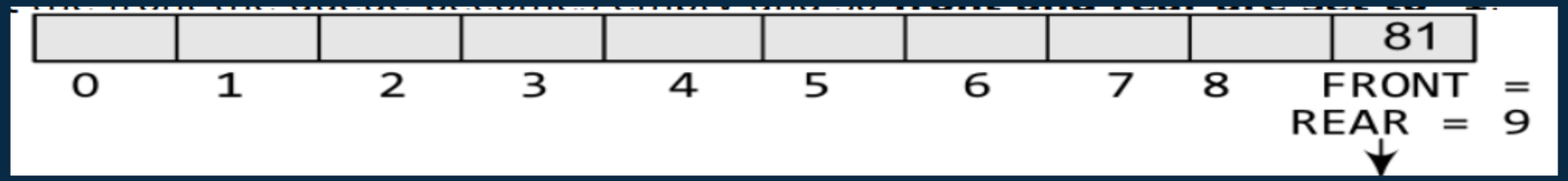
ALGORITHM

```
Step 1: IF FRONT = 0 and Rear = MAX – 1 or REAR = FRONT-1
                                    Write "OVERFLOW”
                        Goto step 4
          [End OF IF]
Step 2: IF FRONT = -1 and REAR = -1
                SET FRONT = REAR = 0
           ELSE  IF REAR = MAX - 1 and FRONT != 0
                SET REAR = 0
           ELSE
                SET REAR = REAR + 1
           [END OF IF]
Step 3: SET QUEUE[REAR] = VAL
Step 4: EXIT
```

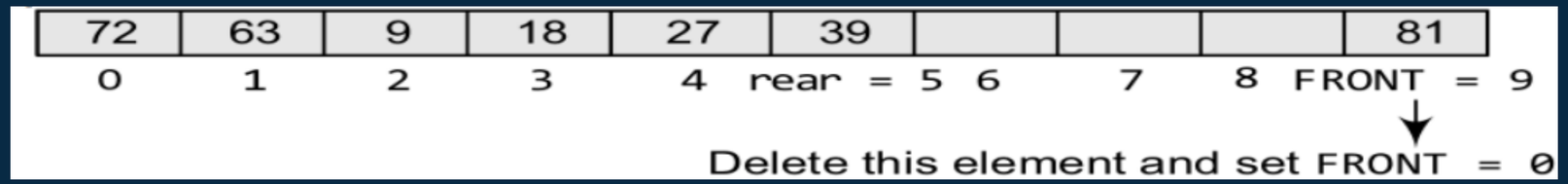
DELETION IN CIRCULAR QUEUE



If front = −1, then there are no elements in the queue. So, an underflow condition.



When front = rear (and both aren't -1), it means the queue has exactly one element.



If the queue is not empty and front = MAX−1, then after deleting the element at the front, front is set to 0.

ALGORITHM

```
Step 1: IF FRONT = -1
            Write "UNDERFLOW"
            Goto Step 4
        [END of IF]
Step 2: SET VAL = QUEUE[FRONT]
Step 3: IF FRONT = REAR
            SET FRONT = REAR = -1
         ELSE
            IF FRONT = MAX -1
                  SET FRONT = 0
            ELSE
                  SET FRONT = FRONT + 1
            [END of IF]
        [END OF IF]
Step 4: EXIT
```

PRIORITY QUEUES

A priority queue is a queue in which each element is assigned a priority. • The priority of elements is used to determine the order in which these elements will be processed.

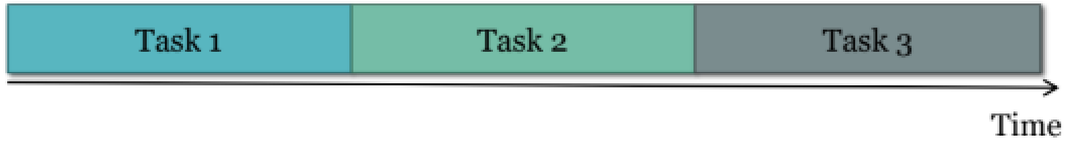
The general rule of processing elements of a priority queue can be given as:

1. An element with higher priority is processed before an element with lower priority.
2. Two elements with same priority are processed on a first come first served (FCFS) basis.

PRIORITY

The first enqueued task of highest priority executes to completion

A task will only relinquish a processor when it completes, yields, or blocks

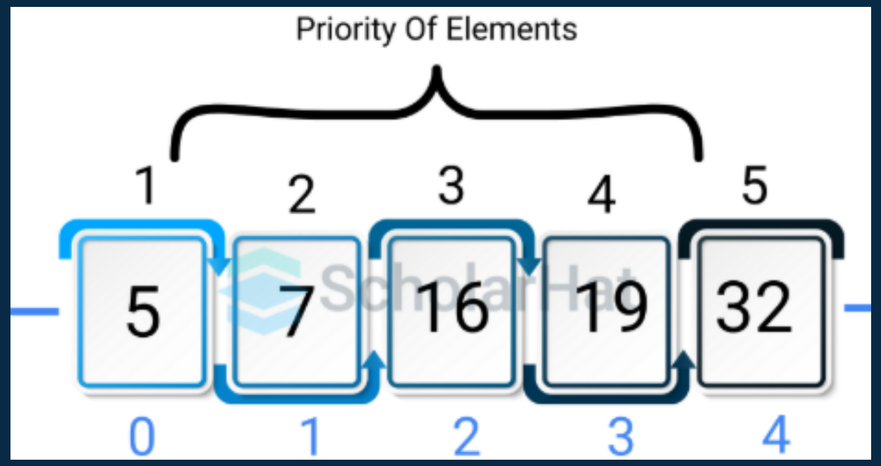


FIFO Real Time Scheduling

IMPLEMENTATION OF PRIORITY QUEUES

SORTED QUEUE

Ascending Order Priority Queue



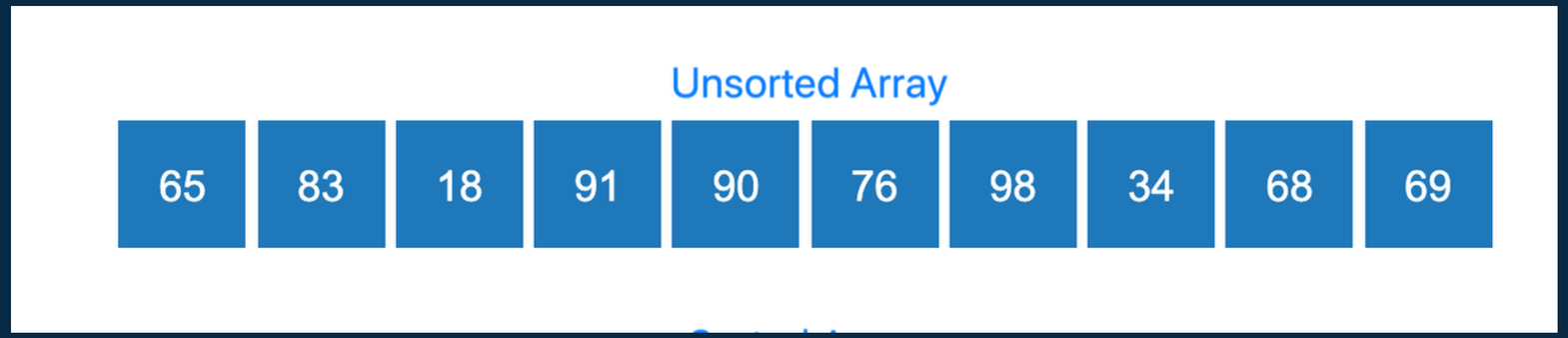
Elements with lower values have higher priority.

DELETION: O(1)

INSERTION: O(n)

https://www.geeksforgeeks.org/priority-queue-set-1-introduction

UNSORTED QUEUE



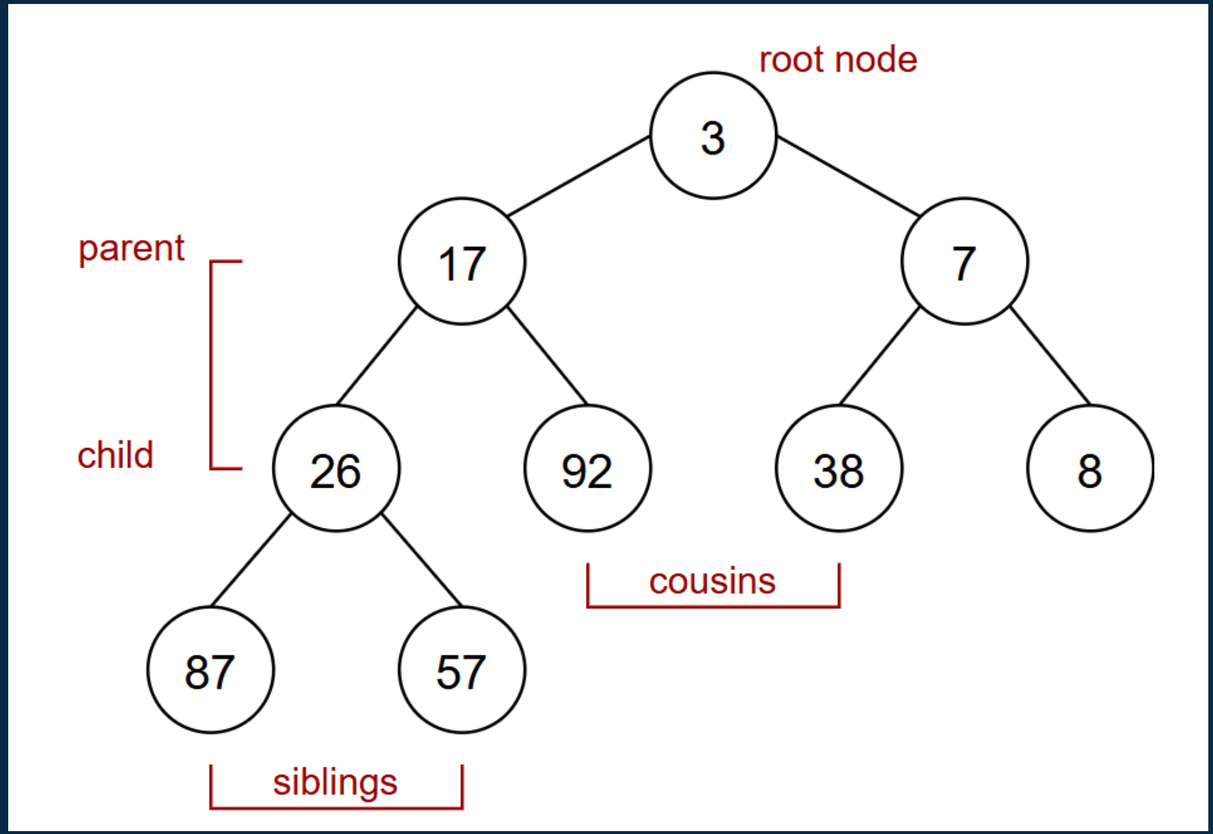
DELETION: O(n)

INSERTION: O(1)

Implement Priority Queue Using Linked List

Linked List	push()	pop()	peek()
Time Complexity	O(n)	O(1)	O(1)

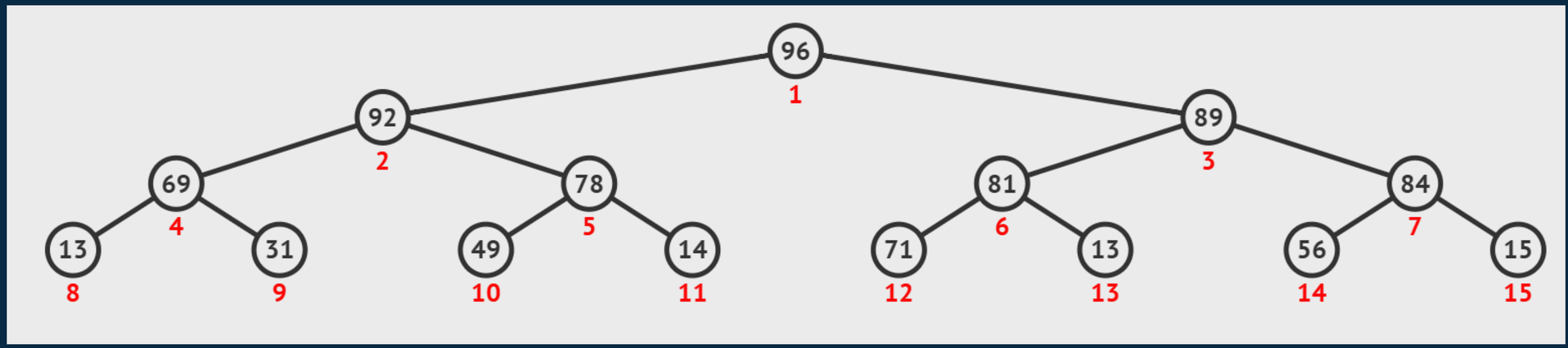
Implement Priority Queue Using Heap



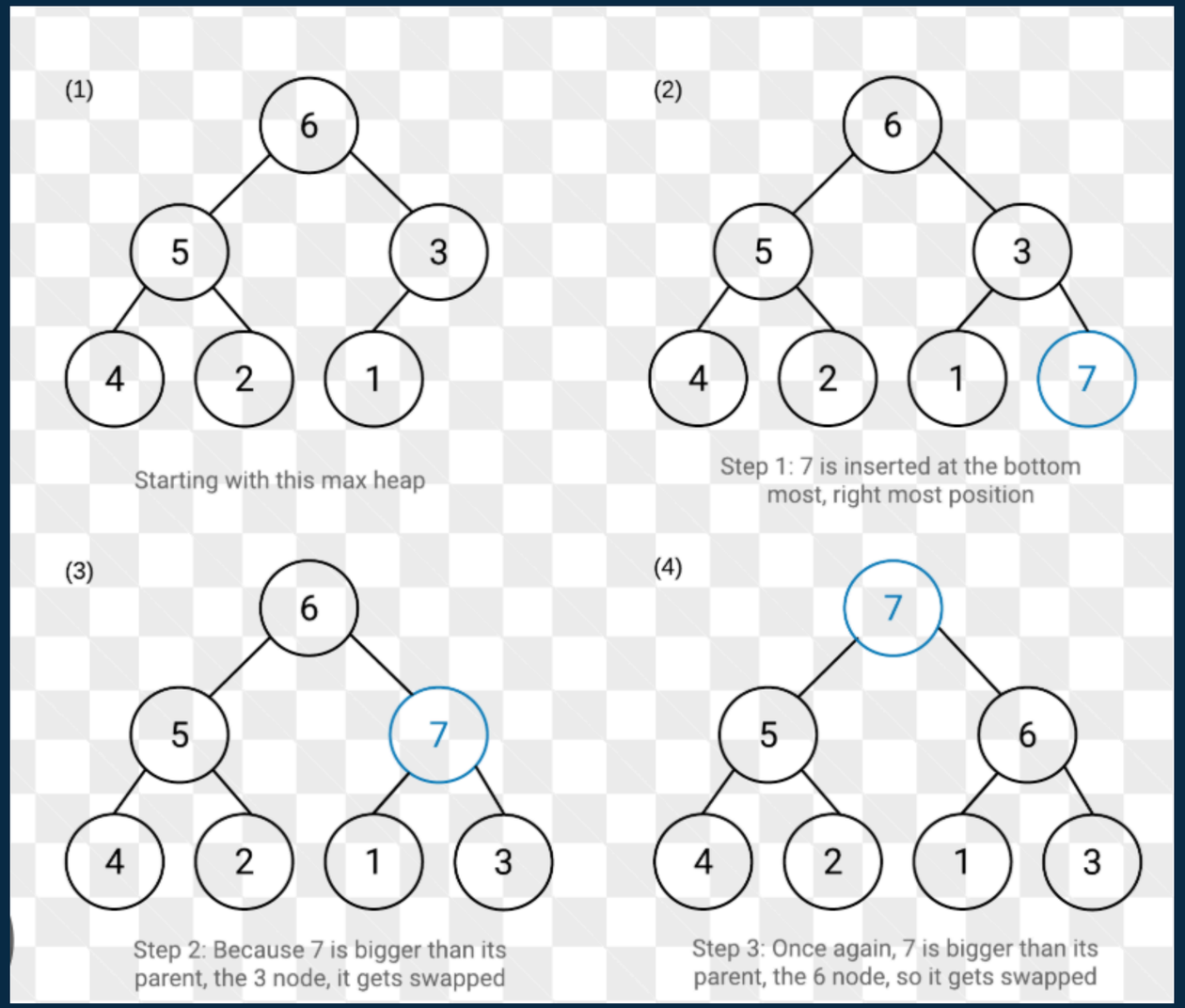
https://www.geeksforgeeks.org/dsa/heap-data-structure/

The height of a height-balanced binary tree with N nodes is O(logN).

Implement Priority Queue Using Heap

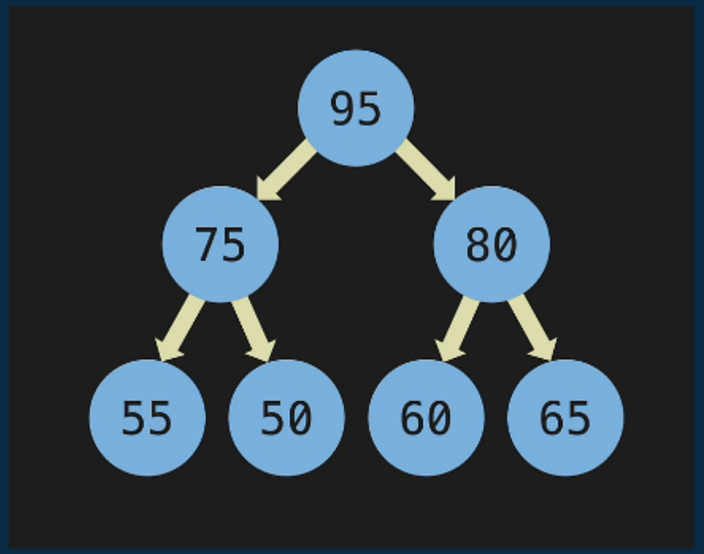


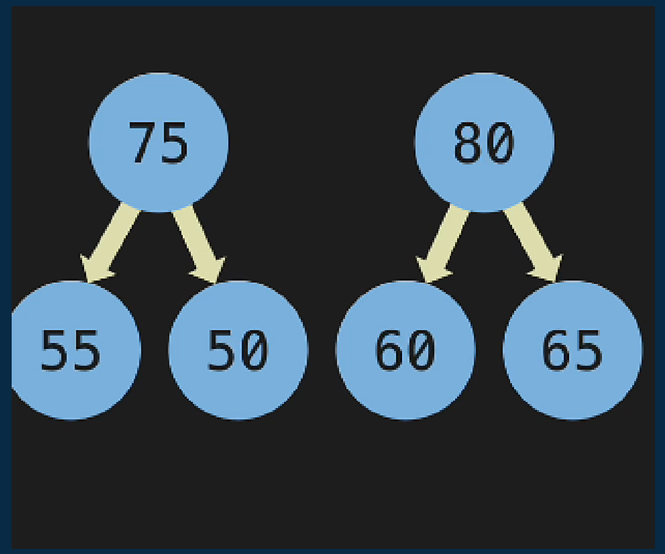
INSERTION IN A HEAP

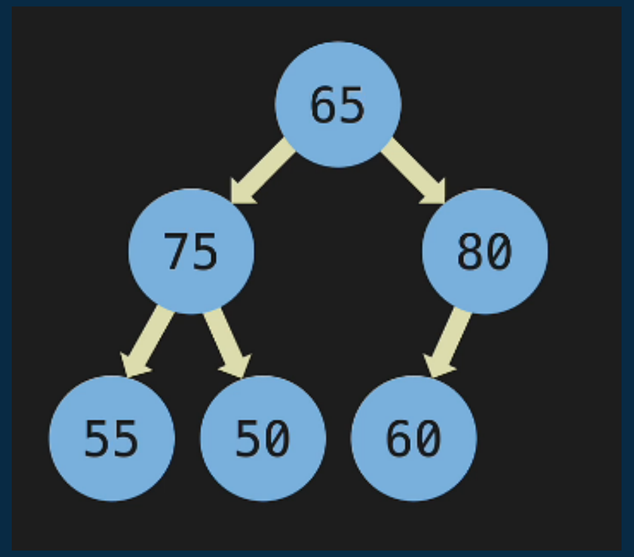


Complexity
O(logN)

DELETION IN A HEAP







Complexity

O(logN)

Then heapify

Due to lesser complexity, priority queues are implemented using heap data struture

APPLICATIONS

CPU Scheduling and Task Management

Breadth-First Search (BFS) and Graph Traversal

Asynchronous Data Handling and Buffering

Handling Interrupts in Operating Systems

https://www.geeksforgeeks.org/problems/queue-using-two-stacks--115418/1?page=2&category=Queue&sortBy=submissions

REFERENCES

Study Materials of Dr. Shweta Saharan

GFG and VisuAlgo

THANK YOU