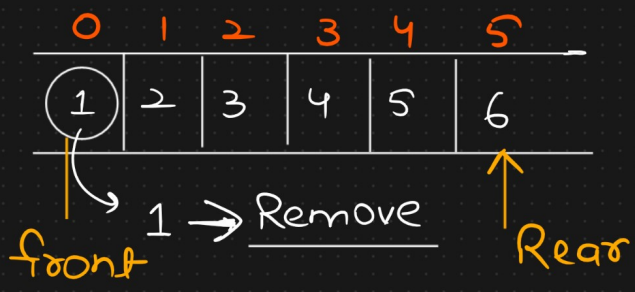
Queue

1) FIFO (First In First Out)



2) front → deletion

Rear → insertion

3) Abstract Data Type

1) Enqueue — Insertion

2) Dequeue — Deletion

4) Real Time Application

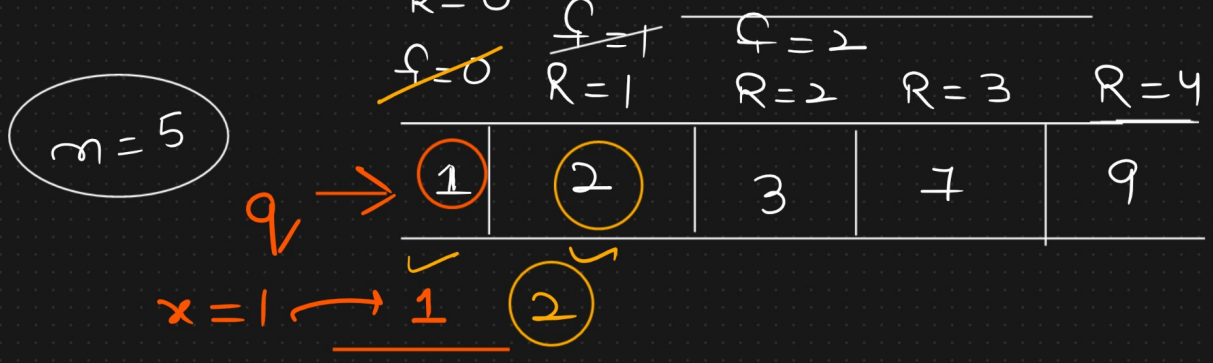
1) Browsing history — FIFO

Queue → FIFO size() → size of the queue

Example peek() → to print the top element in queue

1, 2, 3, 7, 9, ~~12~~

front = Rear = -1



add() ⟷ Enqueue(x){ ← new element

insertion

```
if(rear == n-1){
    printf("Queue overflow");
}
// Queue is empty
else if(front == rear == -1){
    front = front+1;
    rear = rear+1;
    q[rear] = x;
}
else {
    rear = rear+1;
    q[rear] = x;
}
}
```

Time complexity

↳ O(1)

remove() ⟷ Dequeue

Time complexity
↳ O(1)

```
Dequeue() {
    // Queue is empty
    if (front == -1) {
        Print("Queue underflow");
    }
    else {
        x = q[f];   → Deleted element
        → single element is available in queue
        if (front == rear) {
            front = rear = -1;
        }
        else {
            front = front + 1;
        }
        return x;
    }
```