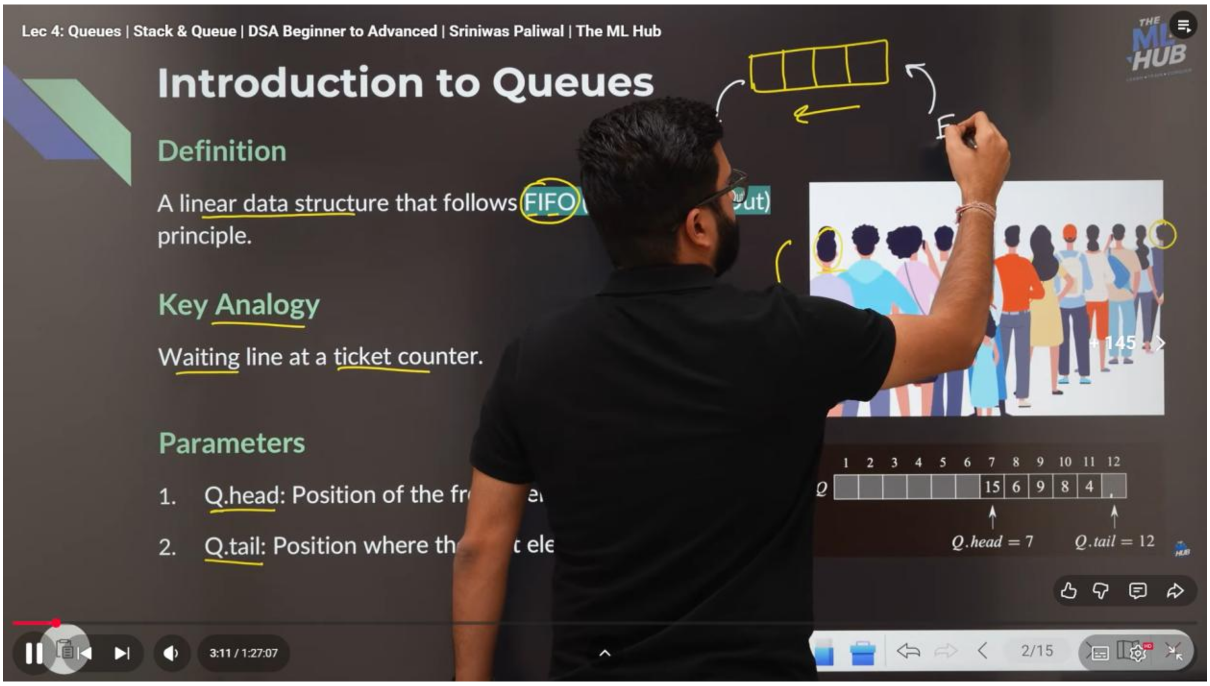
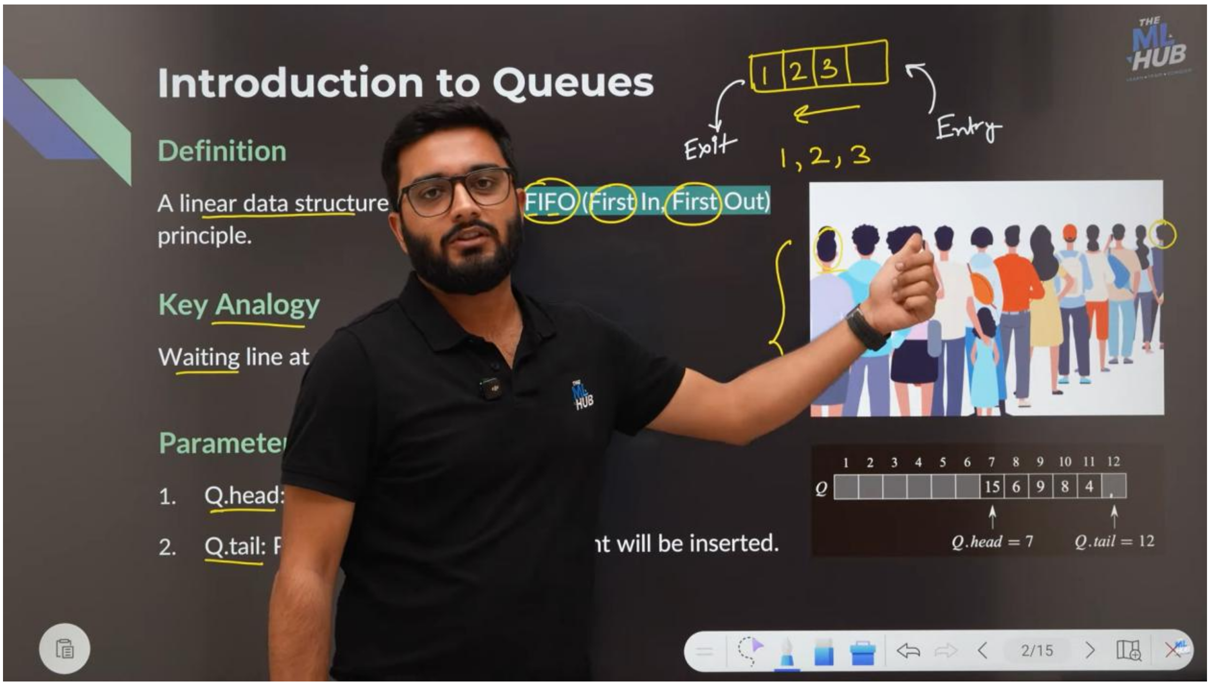
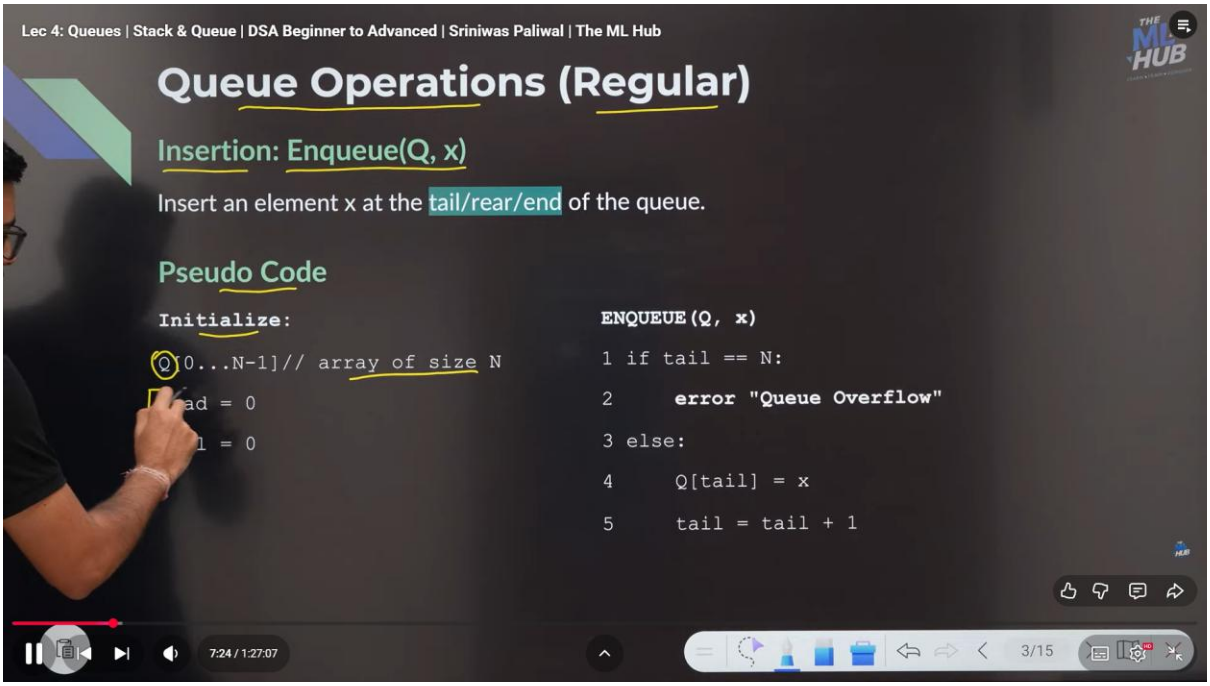
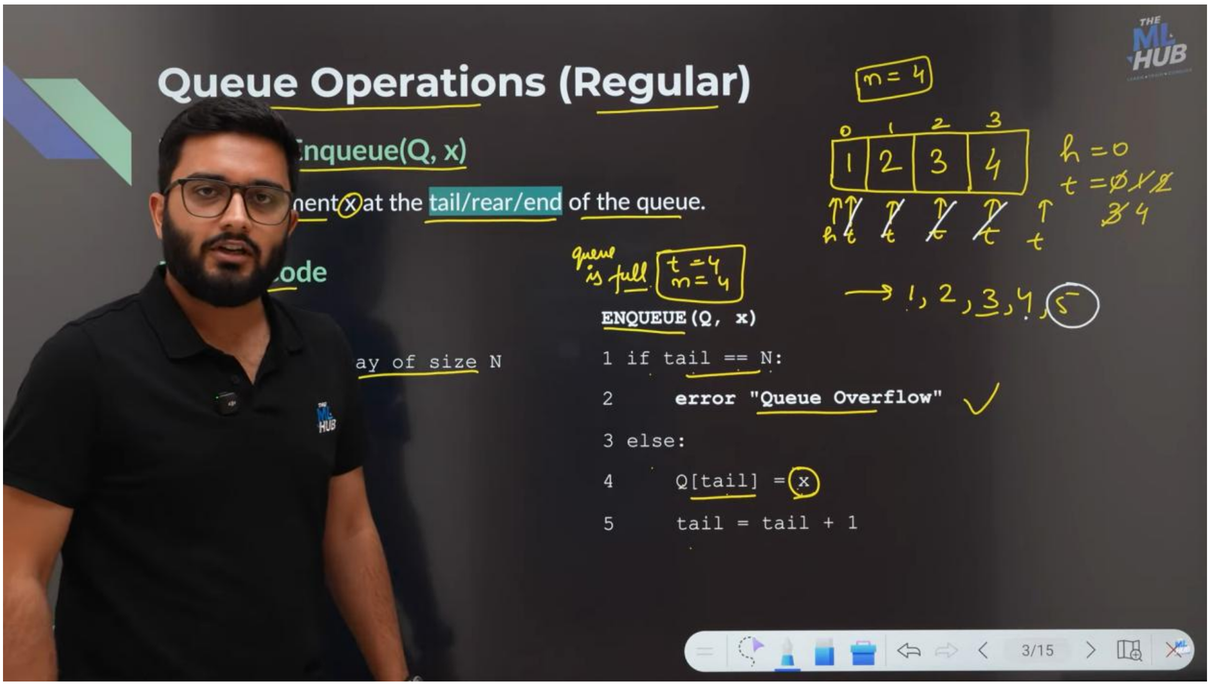


Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Introduction to Queues
Definition
A linear data structure that follows FIFO
principle.
Key Analogy
Waiting line at a ticket counter.
Parameters
1. Q.head: Position of the fr
2. Q.tail: Position where th
Q.head = 7
Q.tail = 12
3:11 / 1:27:07
2/15

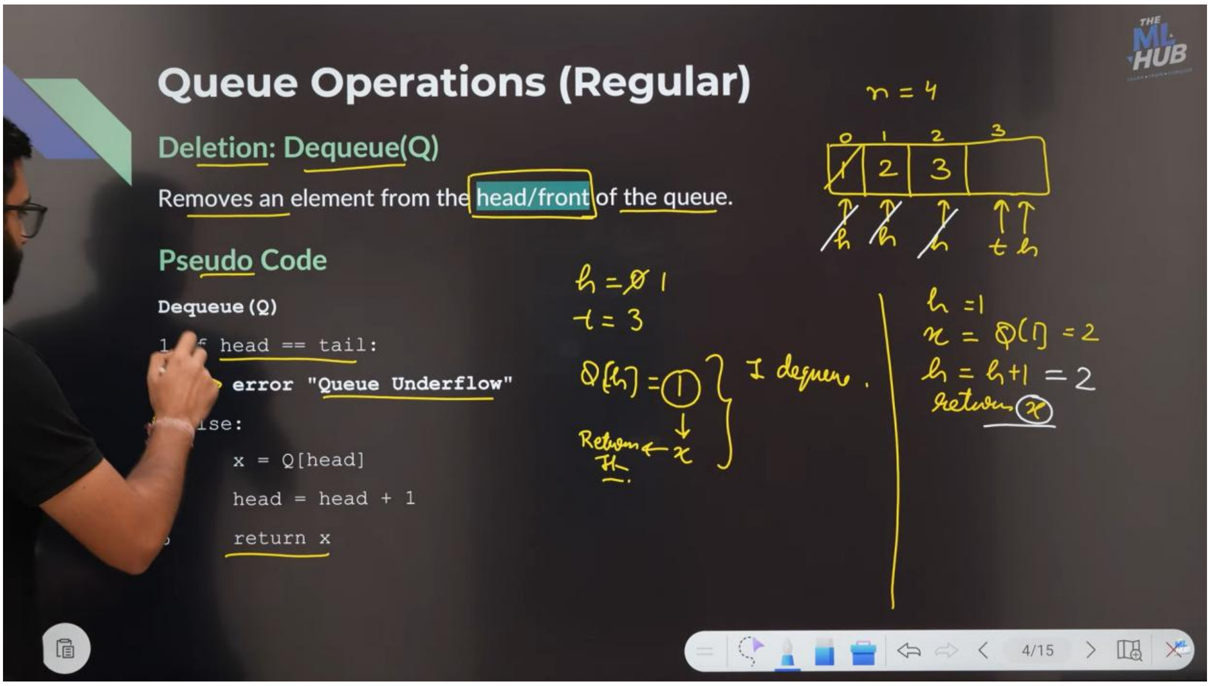


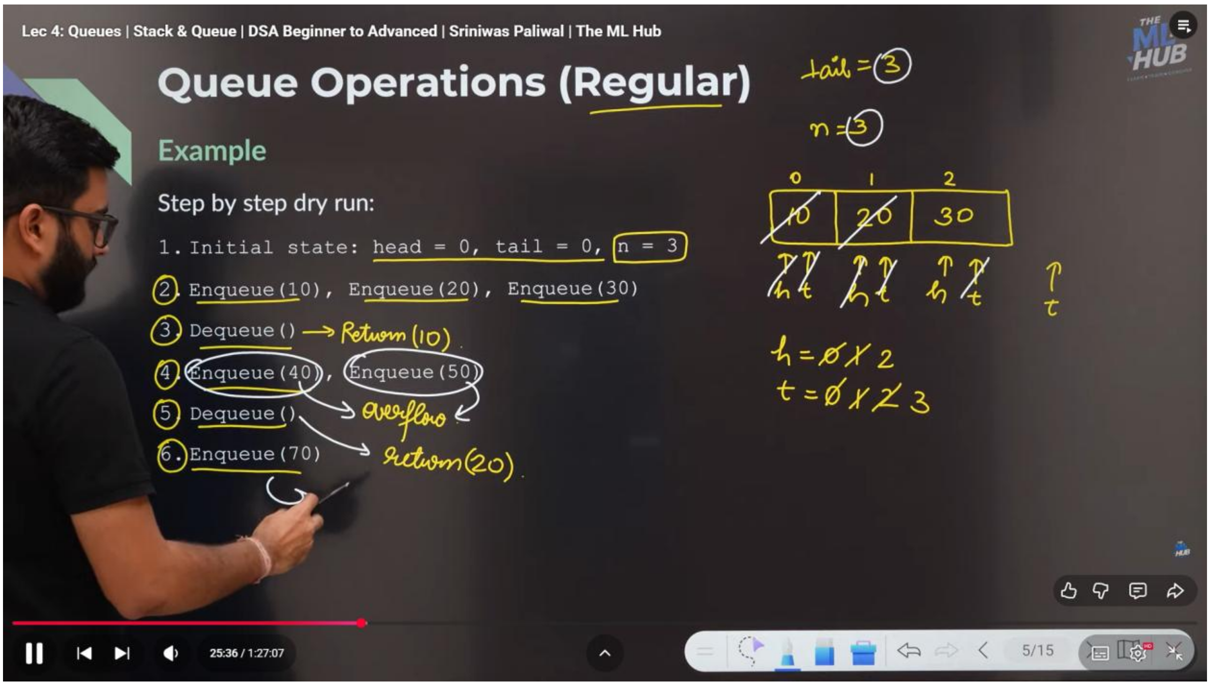
Introduction to Queues
1 2 3
Exit
Entry
1, 2, 3
Definition
A linear data structure
FIFO (First In, First Out)
principle.
Key Analogy
Waiting line at
Paramete
1. Q.head:
2. Q.tail: P
nt will be inserted.
Q.head = 7
Q.tail = 12
2/15

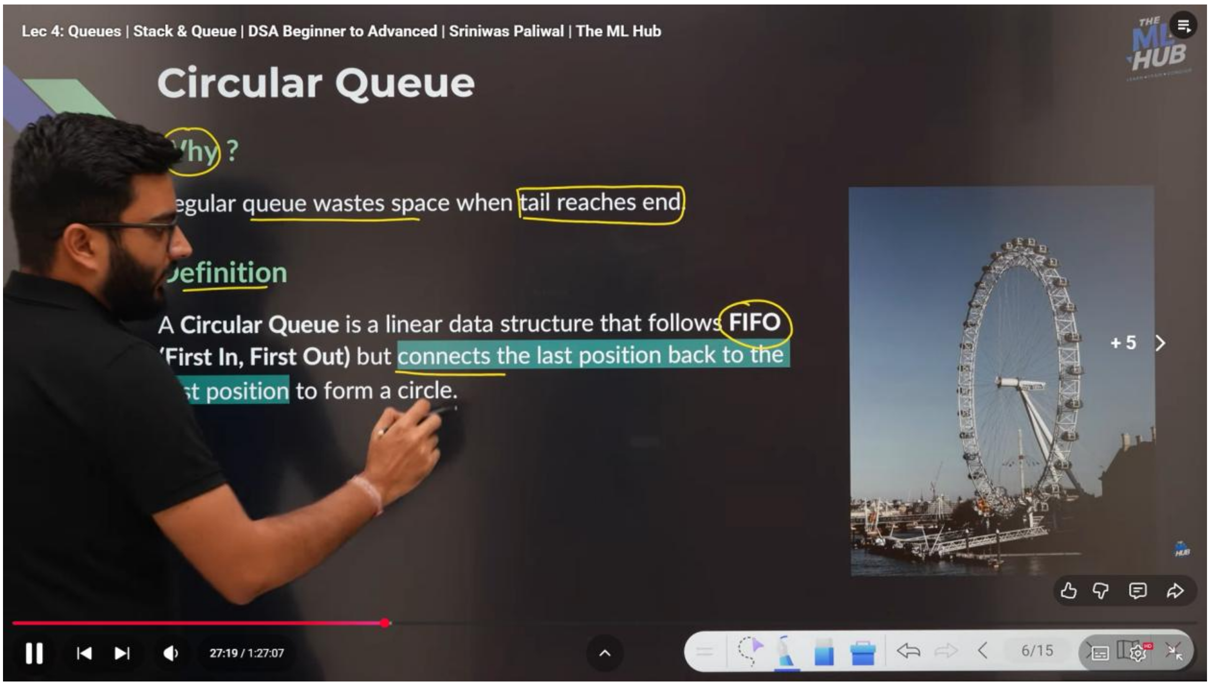


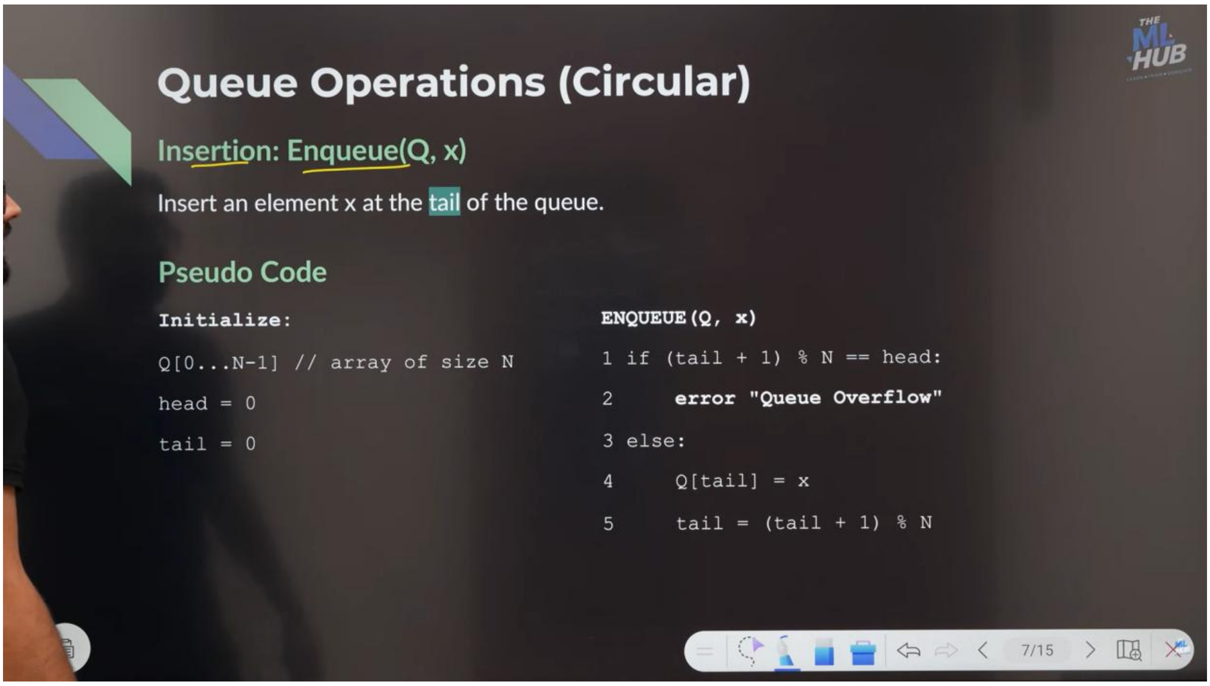


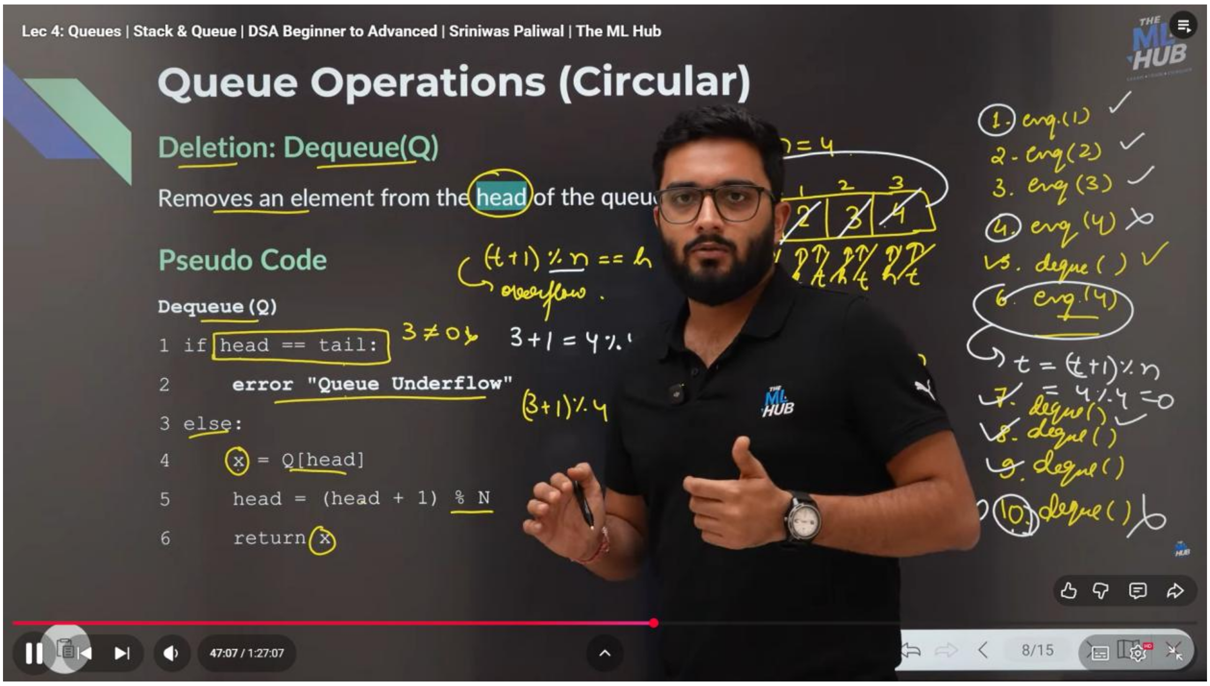
for filling 5 it will be overflow

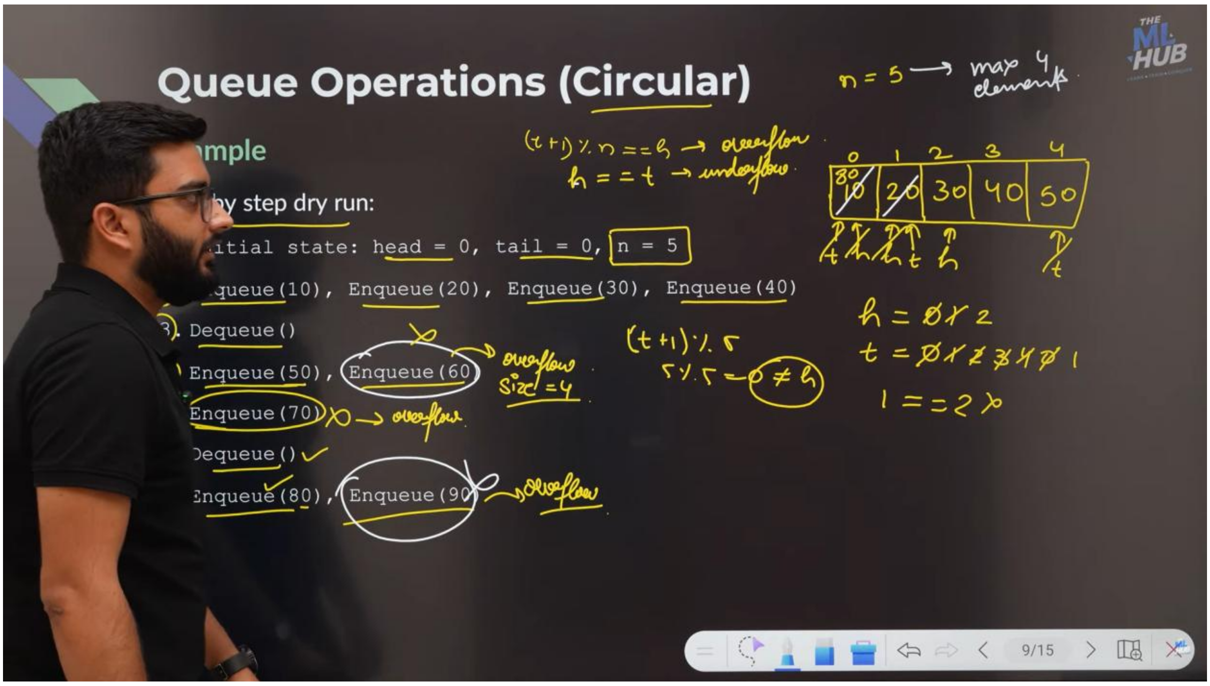

Queue Operations (Regular)
Deletion: Dequeue(Q)
Removes an element from the head/front of the queue.
Pseudo Code
Dequeue(Q)
if head == tail:
error "Queue Underflow"
else:
x = Q[head]
head = head + 1
return x
n = 4
h = 1
x = Q[1] = 2
h = h+1 = 2
return x
4/15

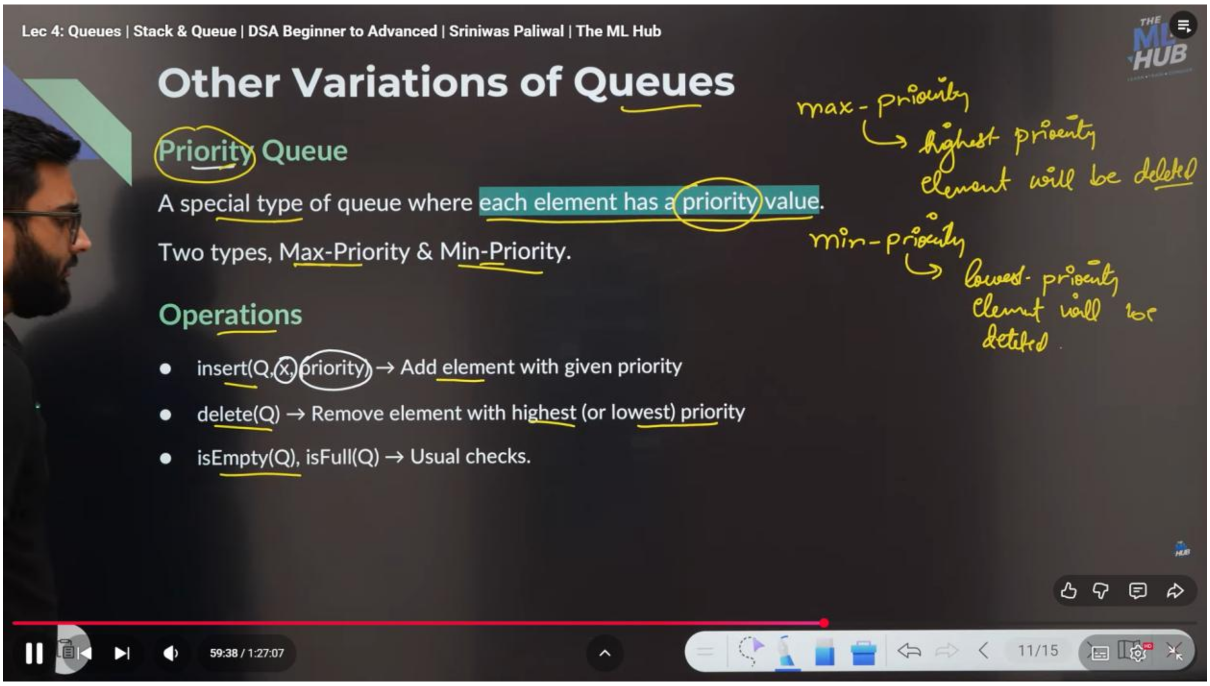

Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Queue Operations (Regular)
Example
Step by step dry run:
1. Initial state: head = 0, tail = 0, n = 3
2. Enqueue(10), Enqueue(20), Enqueue(30)
3. Dequeue() → Return (10)
4. Enqueue(40), Enqueue(50)
5. Dequeue()
6. Enqueue(70)
overflow
return(20)
tail = 3
n = 3
10
20
30
25:36 / 1:27:07
5/15

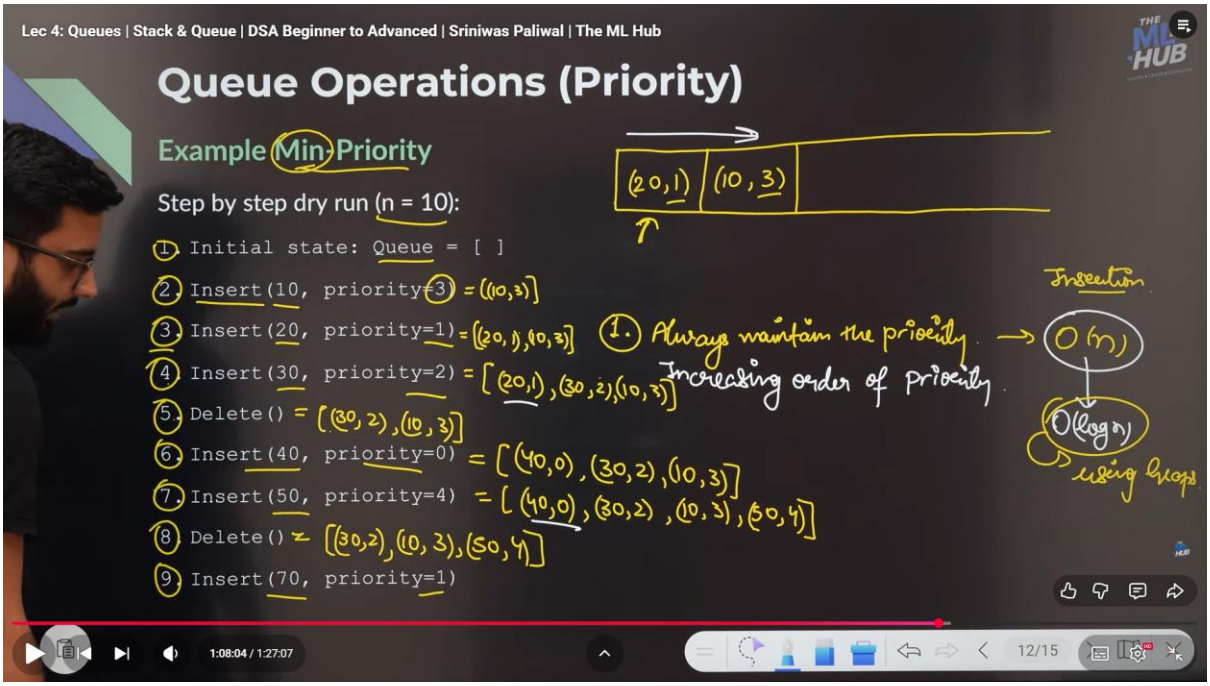

Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Circular Queue
egular queue wastes space when tail reaches end
efinition
A Circular Queue is a linear data structure that follows FIFO
(First In, First Out) but connects the last position back to the
st position to form a circle.
+5
27:19 / 1:27:07
6/15

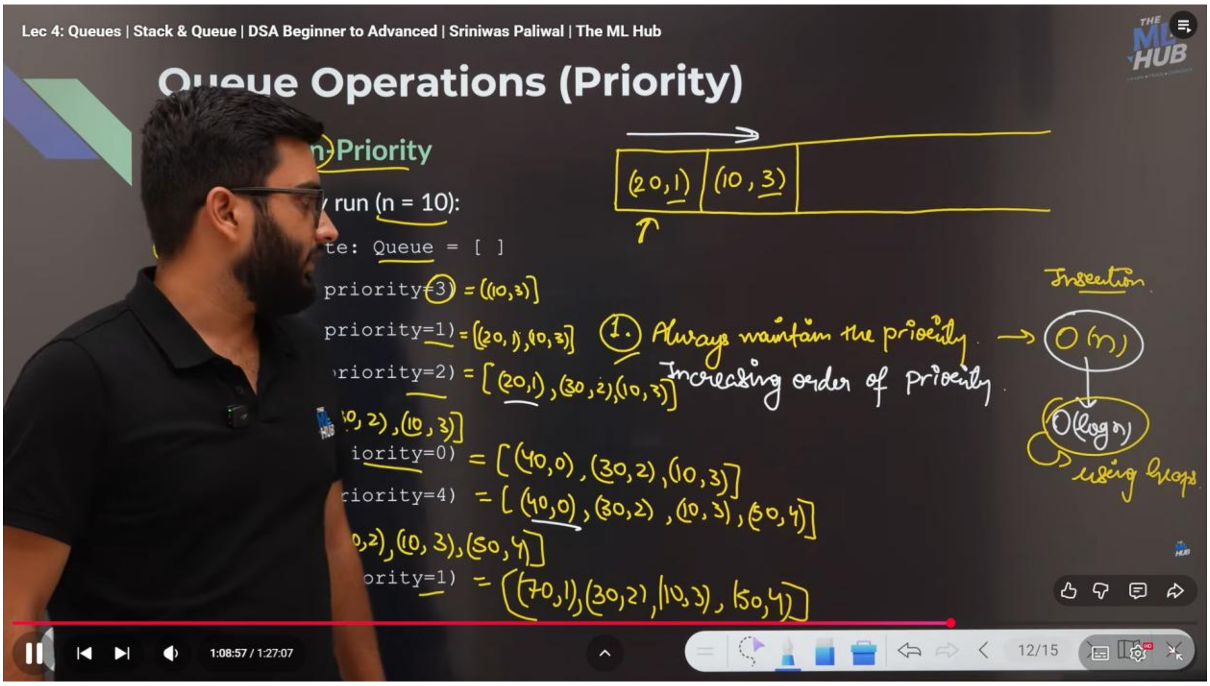

Queue Operations (Circular)
Insertion: Enqueue(Q, x)
Insert an element x at the tail of the queue.
Pseudo Code
Initialize:
Q[0...N-1] // array of size N
head = 0
tail = 0
ENQUEUE(Q, x)
1 if (tail + 1) % N == head:
2 error "Queue Overflow"
3 else:
4 Q[tail] = x
5 tail = (tail + 1) % N
7/15


Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Queue Operations (Circular)
Deletion: Dequeue(Q)
Removes an element from the head of the queu
Pseudo Code
Dequeue(Q)
1 if head == tail:
2 error "Queue Underflow"
3 else:
4 x = Q[head]
5 head = (head + 1) % N
6 return x
(t+1) % n == h
overflow.
1. enq(1)
2. enq(2)
3. enq(3)
4. enq(4)
5. deque()
6. enq(4)
7. deque()
8. deque()
9. deque()
10. deque()
47:07 / 1:27:07
8/15


Queue Operations (Circular)
n = 5 → max 4 elements
mple
by step dry run:
itial state: head = 0, tail = 0, n = 5
queue(10), Enqueue(20), Enqueue(30), Enqueue(40)
Dequeue()
Enqueue(50), Enqueue(60)
Enqueue(70)
Dequeue()
Enqueue(80), Enqueue(90)
(t+1) % n == h → overflow
h == t → underflow
overflow
size = 4
overflow
overflow
9/15


Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Other Variations of Queues
Priority Queue
A special type of queue where each element has a priority value.
Two types, Max-Priority & Min-Priority.
Operations
insert(Q, x, priority) → Add element with given priority
delete(Q) → Remove element with highest (or lowest) priority
isEmpty(Q), isFull(Q) → Usual checks.
max-priority
↳ highest priority element will be deleted
min-priority
↳ lowest priority element will be deleted.
59:38 / 1:27:07
11/15


Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Queue Operations (Priority)
Example Min-Priority
Step by step dry run (n = 10):
(20,1) (10,3)
1. Initial state: Queue = [ ]
2. Insert(10, priority=3) = [(10,3)]
3. Insert(20, priority=1) = [(20,1),(10,3)]
4. Insert(30, priority=2) = [(20,1),(30,2),(10,3)]
5. Delete() = [(30,2),(10,3)]
6. Insert(40, priority=0) = [(40,0),(30,2),(10,3)]
7. Insert(50, priority=4) = [(40,0),(30,2),(10,3),(50,4)]
8. Delete() = [(30,2),(10,3),(50,4)]
9. Insert(70, priority=1)
1. Always maintain the priority → O(n)
Increasing order of priority
Insertion
O(log n)
using heaps
1:08:04 / 1:27:07
12/15


Lec 4: Queues | Stack & Queue | DSA Beginner to Advanced | Sriniwas Paliwal | The ML Hub
Queue Operations (Priority)
Priority
run (n = 10):
te: Queue = [ ]
priority=3) = [(10,3)]
priority=1) = [(20,1),(10,3)]
riority=2) = [(20,1),(30,2),(10,3)]
iority=0) = [(40,0),(30,2),(10,3)]
iority=4) = [(40,0),(30,2),(10,3),(50,4)]
ority=1) = [(70,1),(30,2),(10,3),(50,4)]
(20,1)
(10,3)
1. Always maintain the priority
Increasing order of priority
Insertion
O(n)
O(log n)
using heaps
1:08:57 / 1:27:07
12/15