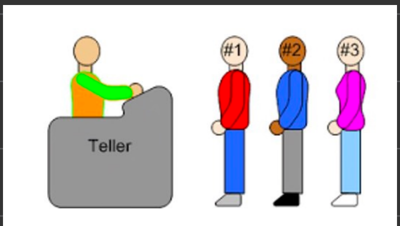
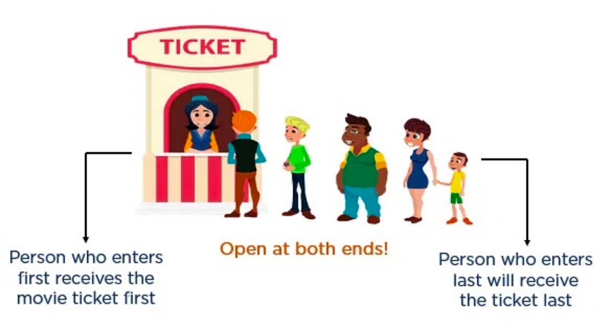
29 June 2023

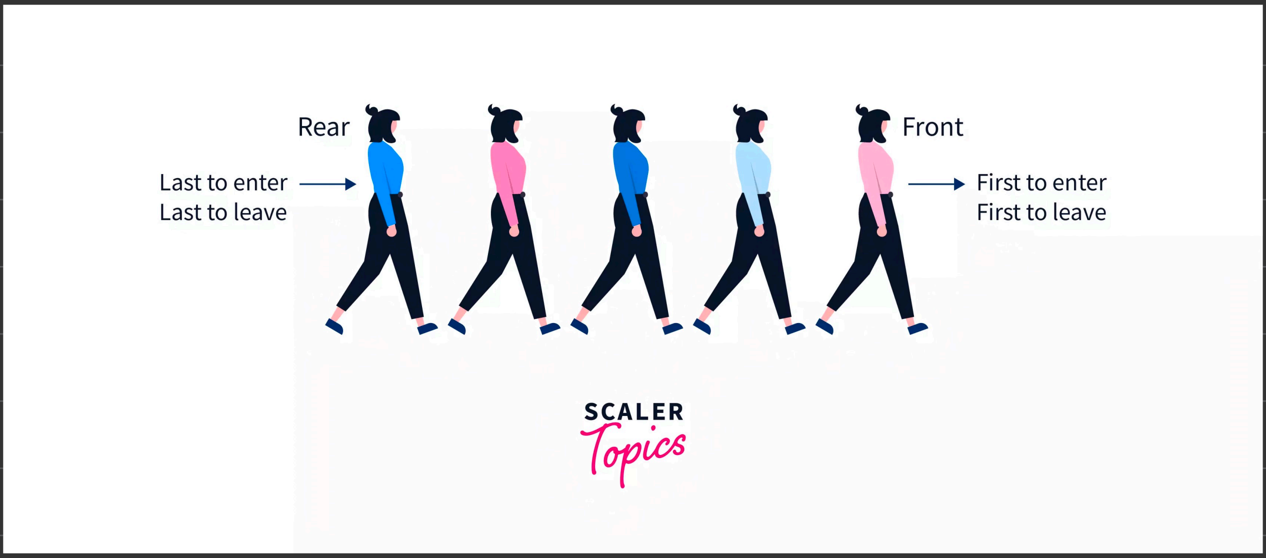
Queue:




TICKET
Person who enters first receives the movie ticket first
Open at both ends!
Person who enters last will receive the ticket last

Hospital Emergency Queue
The person who requires quick medical attention will be treated immediately!

LAST IN LAST OUT
ONE WAY
FIRST IN FIRST OUT



Queue Operations: (FIFO, LILO)

1. Insert into Queue : Enqueue
2 Remove from Queue : Dequeue

	0	1	2	3	4
q	~~10~~	20	40	50	

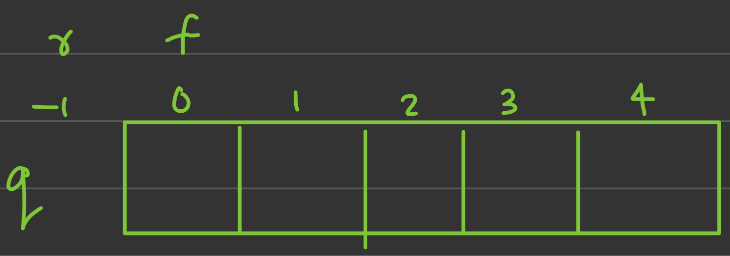
	Enqueue	Dequeue
1	10	
2	20	
3		✓
4	40	
5	50	

Queue Operations:

1. add()
2. peek(): returns head(front) of the queue without removing it
3. element(): Performs the same operation as peek(). Throws NoSuchElementException when the queue is empty
4. remove(): removes the head of the queue and returns it. Throws NoSuchEelementException if the queue is empty
5. poll(): removes the head of the queue and returns it. If the queue is empty, it returns null
6. offer(): insert the new element e into the queue without violating capacity restrictions.
7. size()

Queue using Arrays :

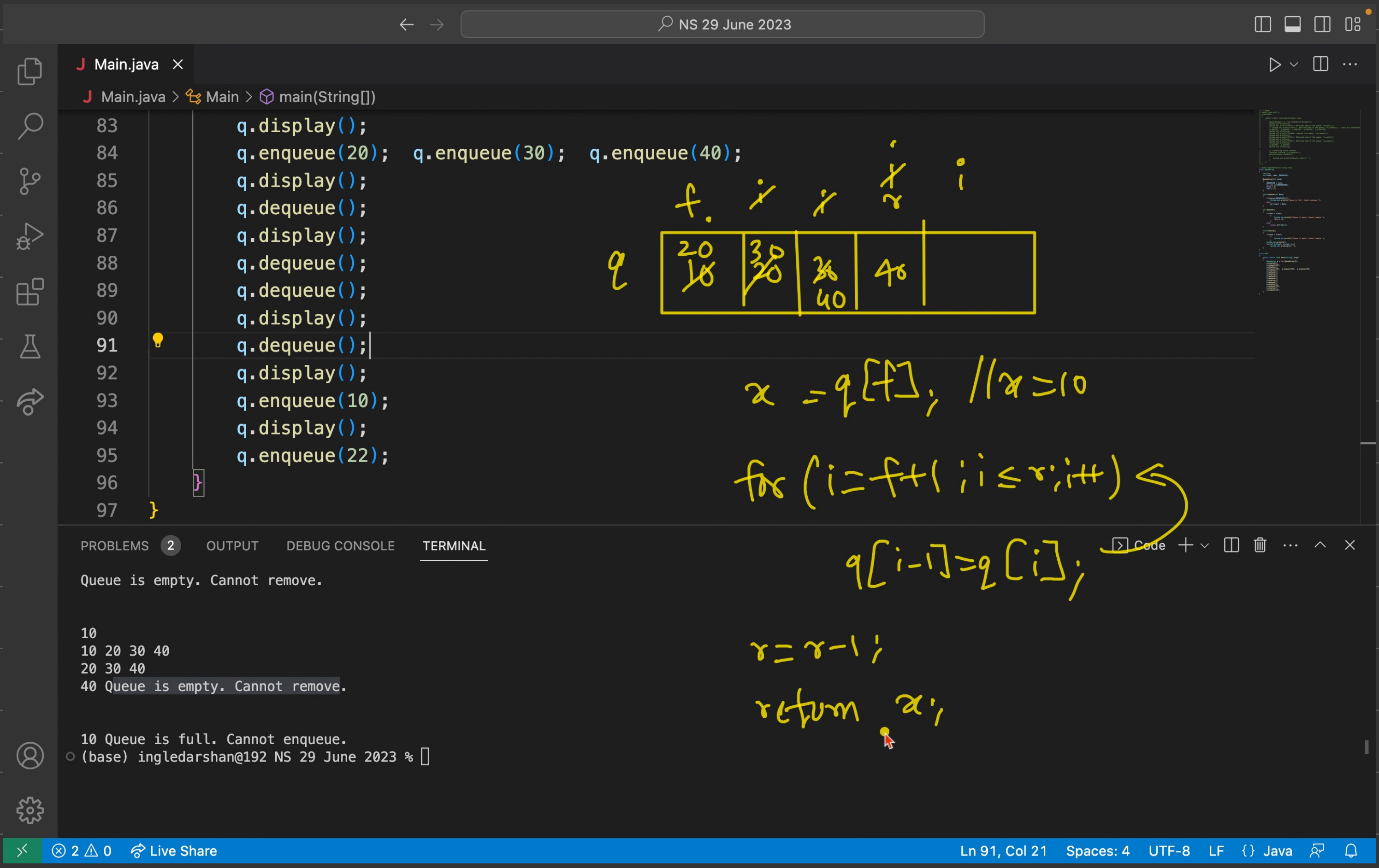
front - f , rear - r



QUEUESIZE = 5

size=0

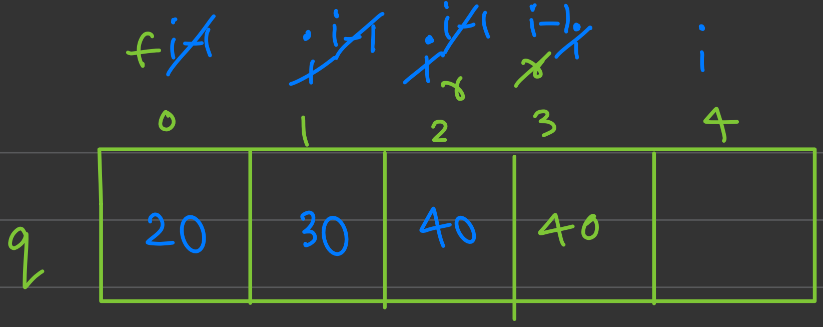
```
enqueue: 1. Insert  data=10
         2  If   rear==(QUEUESIZE -1)
                 Sop(" Queue full Cannot enqueue.")

         3.   q[++rear]= data

dequeue: 1. If  rear <front
                Sop ("Queue Underflow. Cannot remove.")

         2.  return   q[front++];
```



```java
        q.display();
        q.enqueue(20);  q.enqueue(30);  q.enqueue(40);
        q.display();
        q.dequeue();
        q.display();
        q.dequeue();
        q.dequeue();
        q.display();
        q.dequeue();
        q.display();
        q.enqueue(10);
        q.display();
        q.enqueue(22);
    }
}
```

```
x =q[f]; //x=10

for (i=f+1 ;i≤r;i++)

    q[i-1]=q[i];

r=r-1;
return x;
```

```
Queue is empty. Cannot remove.


10
10 20 30 40
20 30 40
40 Queue is empty. Cannot remove.


10 Queue is full. Cannot enqueue.
(base) ingledarshan@192 NS 29 June 2023 % 
```



Remove / Deque:

```
int x = q[f]; //x=10

for (int i= f+1 ;i<=r;i++)
        q[i-1] = q[i]

r= r-1;
return x;
```

Traverse:

```
for(i=f; i<=r;i++)
      Sop(q[i])
```