Q1.Explain Queue Data Structure in java programming?

Ans: Like Stack Queue is also linear data structure, where first in first out order is required then we should go for Queue Data Structure

In Queue Insertion perform at rear END and deletion perform in front end

Operation of Queue

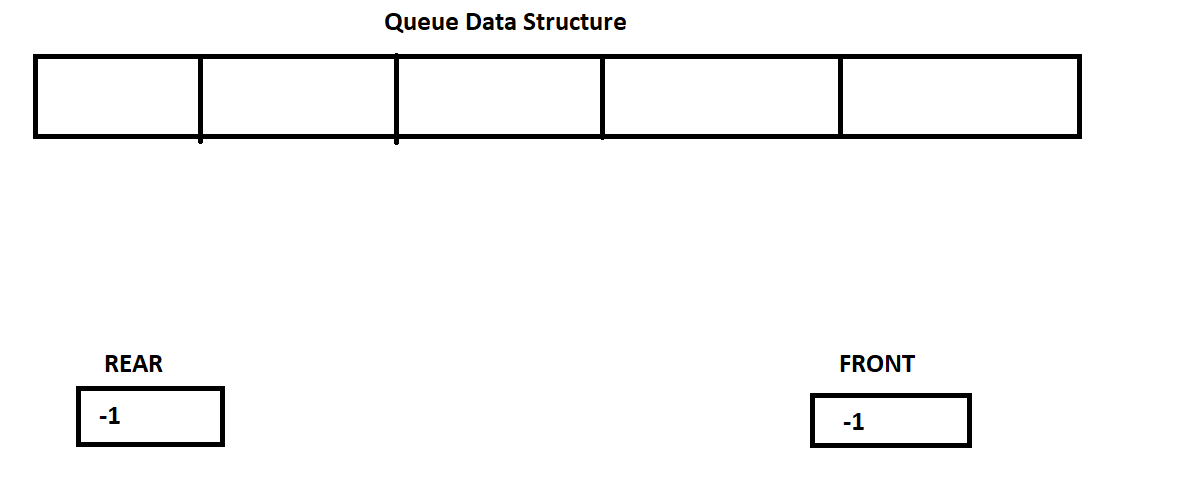
1. enque(): Insert data into the queue(REAR)
2. Deque(): delete the data from the queue(FRONT)
3. Peek(): It always return front element of the Queue
4. isEmpty(): To check queue is empty or not
5. isFull(): To check queue is Full or not

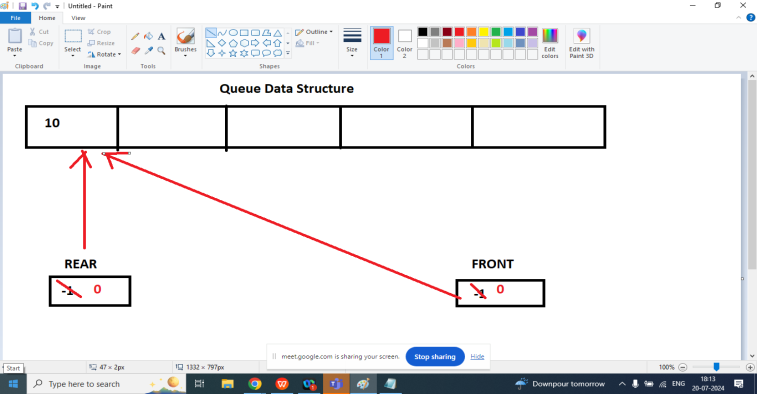
Queue Implementation using Array

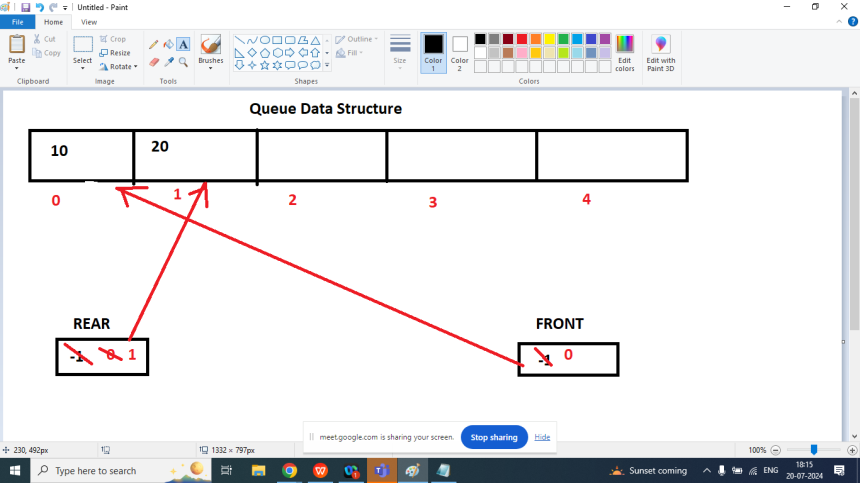
int q[]=new int[5];

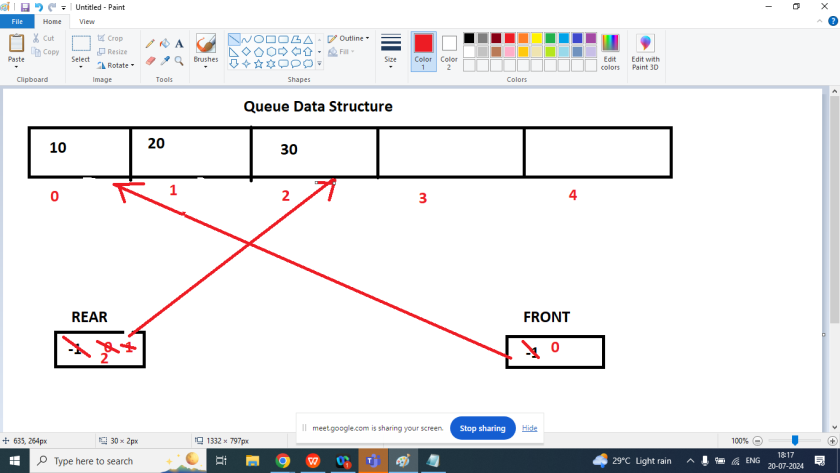
int front=-1;

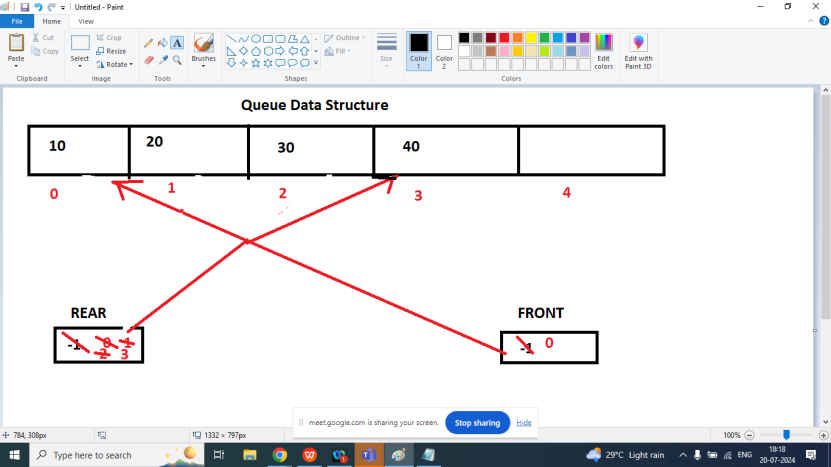
Int rear=-1;

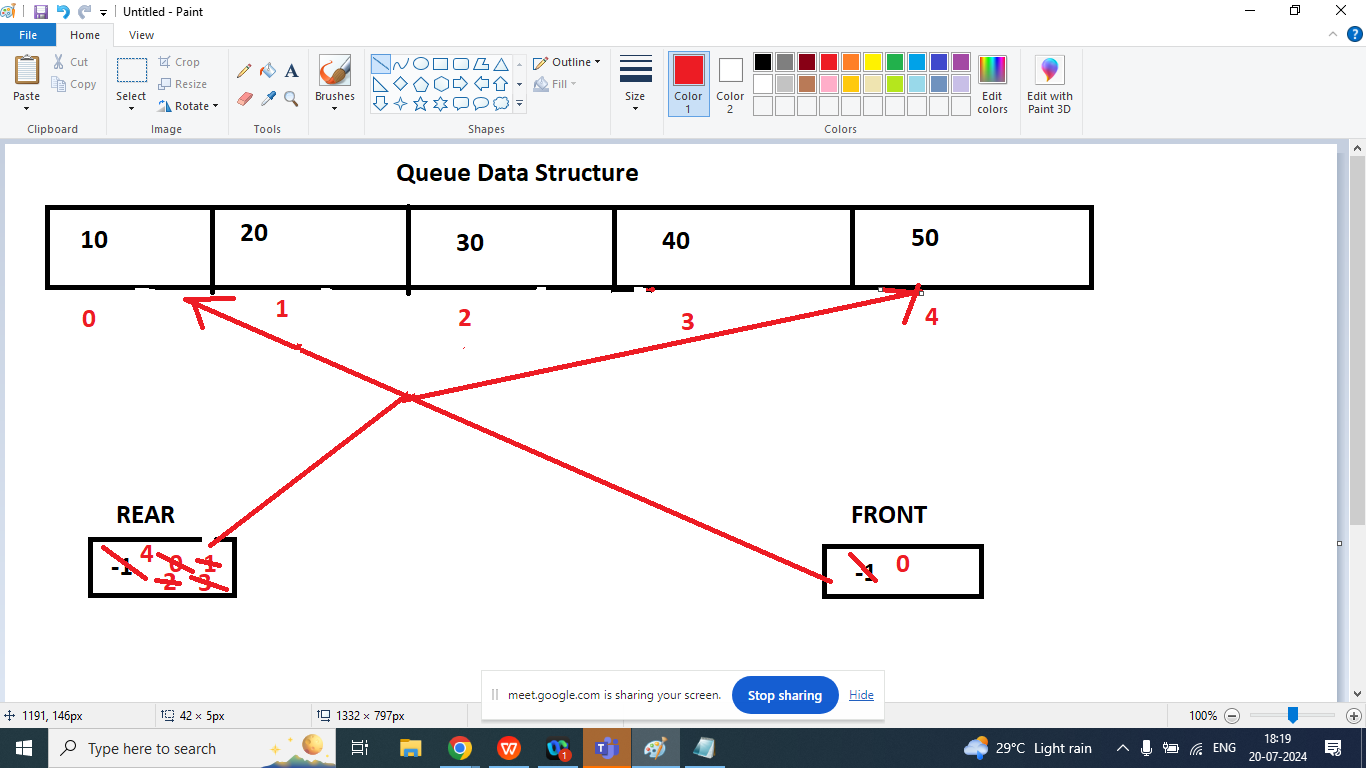












class QueueDemo{

int q[]=new int[5];

int front=-1;

int rear=-1;

public void enq(int data){

if(rear==q.length-1){

System.out.println("This is Over flow condition");

}

else{

if(front==-1 && rear==-1){

front++;

rear++;

q[rear]=data;

}

else{

rear++;

q[rear]=data;

}

}

}

public void display(){

if(front==-1&&rear==-1){

System.out.println("Queue is Empty ");

}

else{

System.out.println("Queue Elements Are : ");

for(int i=front;i<=rear;i++){

System.out.print("\t"+q[i]);

}

}

}

public int deq(){

int r=-1;

if(front==-1&&rear==-1){

System.out.println("Under Flow ");

}

else{

if(front==rear){

r=q[front];

front=rear=-1;

}

else{

r=q[front];

front++;

}

}

return r;

}

public int peek(){

if(front==-1&&rear==-1){

return -1;

}

return q[front];

}

public static void main(String args[]){

QueueDemo qd=new QueueDemo();

qd.enq(10);

qd.enq(20);

qd.enq(30);

qd.enq(40);

qd.enq(50);

qd.display();

System.out.println("\nDeleted Element : "+qd.deq());

System.out.println("\nDeleted Element : "+qd.deq());

System.out.println("\nDeleted Element : "+qd.deq());

System.out.println("\nFront Element : "+qd.deq());

System.out.println("\nDeleted Element : "+qd.deq());

System.out.println("\nDeleted Element : "+qd.deq());

System.out.println("\nDeleted Element : "+qd.deq());

qd.display();

}

}