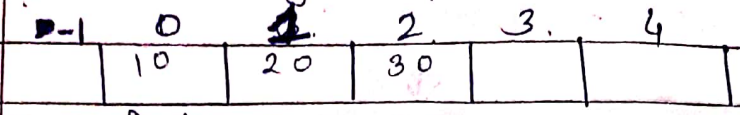
Page No.
Date

Queue using oop concepts in C++



// initially.

rear = -1 front = 0
count = 0

count
0, 1, 2, 3,
2

main()
queue q; // creating object of class queue

front
0, 1

* q.enqueue(10);
↓
enqueue(int item);
↓
if(isfull())
↓
return (size() == capacity)
0 == 3 false
inserting item = item;
= 10.

rear
-1, 0, 1, 2

rear = (rear + 1) % capacity;
rear = -1 + 1 % 3.
= 0 % 5
= 0
count ++

arr[rear] = item;
[0] = item
[0] = 10;
↑
10 is inserted

Page No.
Date

* q.enqueue(20);
  ↓
  enqueue(int item);
  ↓
  if(isfull())
  ↓
  return (size() == capacity
  1 == capacity false
  ↵
  inserting item = item
  = 20
  rear = (rear + 1) % capacity
  = (0 + 1) % 5
  = 1
  count ++; Hence 20 inserted

* q.enqueue(30);
  ↓
  enqueue(int item);
  ↓
  if(isfull())
  ↓
  return (size() == capacity
  2 == capacity false
  inserting item = item
  = 30
  rear = (rear + 1) % capacity
  = (1 + 1) % 5
  = 2
  count ++ Hence 30 inserted

③

Page No.
Date

front element = q.peek();

↓

if (isEmpty())

↓

size() == 0

↓

size() = capicity

↓

return count

↓

3

3 == capicity ⇒ false

return arr[front] ← 80

arr[0]

↓

10 ← Hence front element is

10

q.dequeue

↓

if (isEmpty())

↓

size() == 0

↓

size() == capicity

↓

return count

↓

3 == capacitiy ⇒ false

← 80

Removing element

front = (front + 1) % capicity;

count--;

front = (0+1) % 3
= 1

Removing element => arr[front] = [1]
↓
10. ← 10 is removed

* q.size()

Queue size is:-

size()
↓
return count;
↓
count = 2.
↑
Queue size is 2

* if (q.isEmpty())
↓
return (size() == 0);
↳ return count
↳ count = 2
∴ Queue size is 2.

Queue is Not Empty.