0641 Design Circular Deque(LinkedList)

18 November 2024 03:55

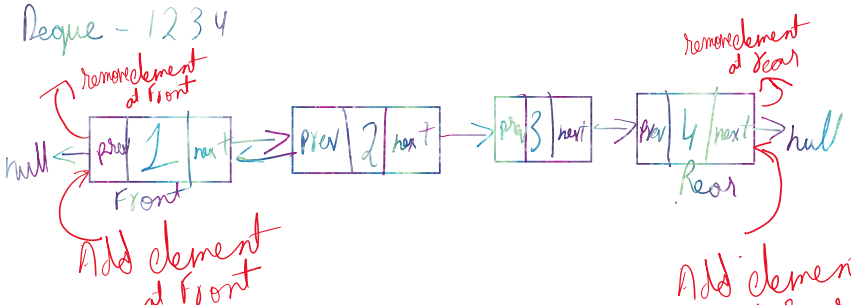
Design your implementation of the circular double-ended queue (deque).

Implement the `MyCircularDeque` class:

- `MyCircularDeque(int k)` Initializes the deque with a maximum size of `k`.
- `boolean insertFront()` Adds an item at the front of Deque. Returns `true` if the operation is successful, or `false` otherwise.
- `boolean insertLast()` Adds an item at the rear of Deque. Returns `true` if the operation is successful, or `false` otherwise.
- `boolean deleteFront()` Deletes an item from the front of Deque. Returns `true` if the operation is successful, or `false` otherwise.
- `boolean deleteLast()` Deletes an item from the rear of Deque. Returns `true` if the operation is successful, or `false` otherwise.
- `int getFront()` Returns the front item from the Deque. Returns `-1` if the deque is empty.
- `int getRear()` Returns the last item from Deque. Returns `-1` if the deque is empty.
- `boolean isEmpty()` Returns `true` if the deque is empty, or `false` otherwise.
- `boolean isFull()` Returns `true` if the deque is full, or `false` otherwise.

Using Link List

Deque - 1 2 3 4



allowed insertion and deletion from both end

We have size, count (int)

Node → rear, front (ref)

a Node

① on initialization

Set count = 0, size = k

and rear, front = null;

② Insert front Let k = 7

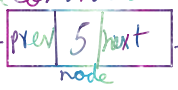
first check whether queue is full or not.

if full return false

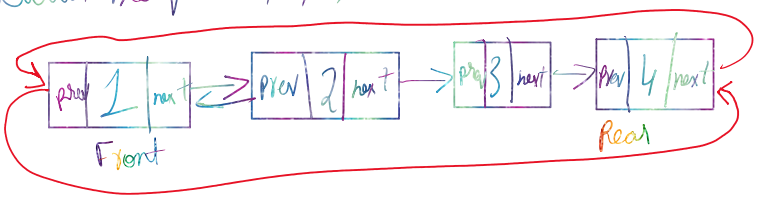
else

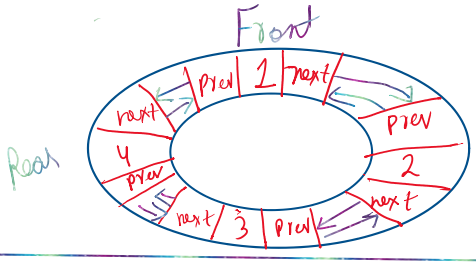
Suppose we have to insert 5

Call Node constructor to make one node

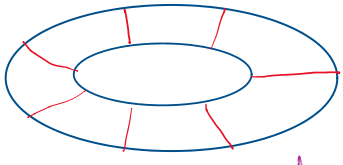


Circular Deque 1, 2, 3, 4





if count = 0 ie empty



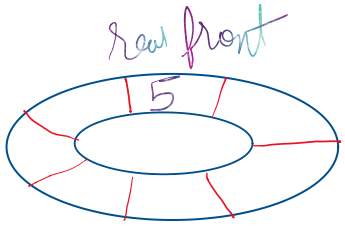
Rear = null

front = null

ie Rear = null and front = null

we will

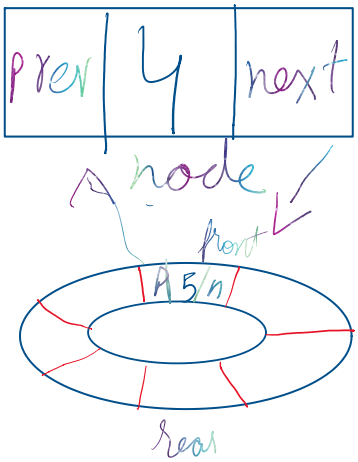
front = node, rear = node



→ if we insert in future to front

else Insert 4

count = 1

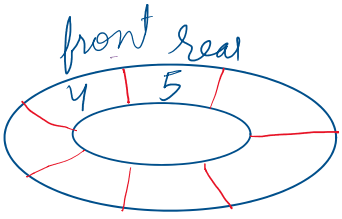


Then front.prev = node

node.next → front

node.prev → null (default)

Front = node

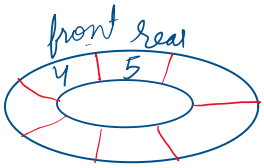


count ++; -2

③ Insert to rear k = 4

Now count = 2

current

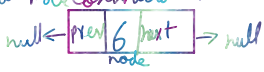


a) if first check whether queue is full or not
if full return false

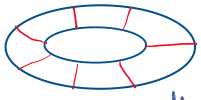
b) else

Suppose we have to insert 6

call Node constructor to make one node



if count = 0 i.e empty

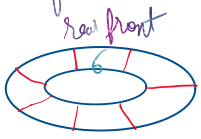


rear = null
front = null

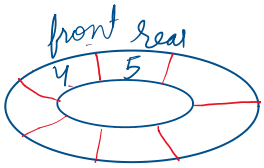
c) rear = null and front = null

we will

front = node, rear = node



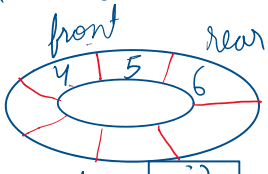
else



⇒ if we insert in future to

to front	to rear

node.next → null (def)
node.prev → rear
rear.next → node
rear = node



Count++; = 3

④ isFull()

check size == count

Yes → Full

No → Not full

⑤ isEmpty()

check count == 0

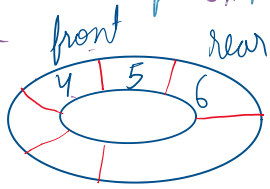
Yes → Empty

No → Not empty

⑥ get front()

if isEmpty() return

else

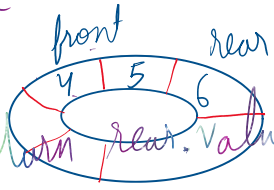


return front.value;

⑦ get rear()

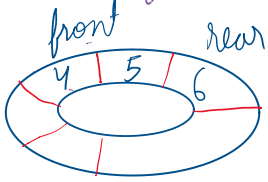
if isEmpty() return -1

else



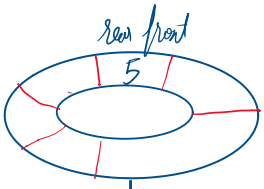
return rear.value;

⑧ delete front



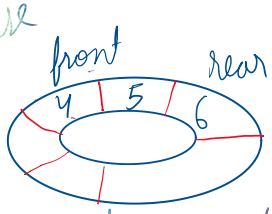
a) if isempty()
return false;

② else if
rear == front i.e 1 node



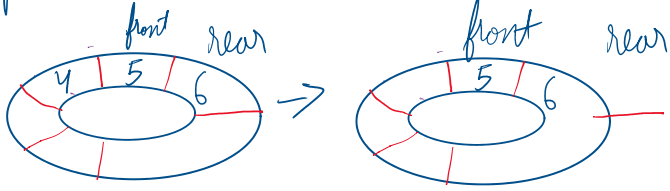
rear = null
front = null

③ else



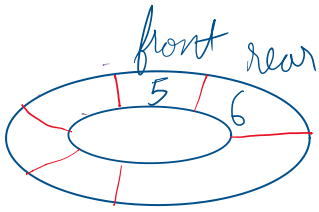
front = front.next

front.prev = null



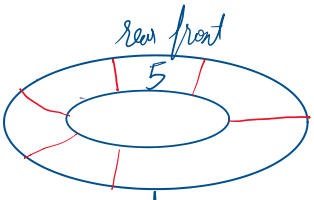
Count--; = 2

9 delete rear



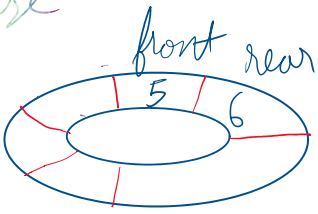
a) if isempty()
return false;

2 else if
rear == front i.e 1 node



rear = null
front = null

3) else



rear = rear.prev
rear.next = null
count--; = 1