PRACTICE PROGRAMMING ASSISGNMENTS **NOT GRADED**
PROF. NAVARTI SAXENA
DATA STRUCTURES AND ALGORITHMS

Queue

1. Design a Circular Queue

Design implementation of a circular queue. The circular queue is a linear data structure in which the operations are performed based on FIFO (First-In-First-Out) principle and the last position is connected back to the first position to make a circle.

One of the benefits of the circular queue is that we can make use of the spaces in front of the queue. In a normal queue, once the queue becomes full, we cannot insert the next element even if there is a space in front of the queue. But using the circular queue, we can use the space to store new values.

Implementation should support following operations:

- MyCircularQueue(k): Set the size of the queue to be k.
- Front: Get the front item from the queue. If the queue is empty, return -1.
- Rear: Get the last item from the queue. If the queue is empty, return -1.
- enQueue(value): Insert an element into the circular queue. Return true if the operation is successful.
- deQueue(): Delete an element from the circular queue. Return true if the operation is successful.
- isEmpty(): Checks whether the circular queue is empty or not.
- isFull(): Checks whether the circular queue is full or not.

Note:

- All values will be in the range of [0, 1000].
- The number of operations will be in the range of [1, 1000].
- Please do not use the built-in Queue library.

https://leetcode.com/explore/learn/card/queue-stack/228/first-in-first-out-data-structure/1337/