class MyCircularDeque:

def \_\_init\_\_(self, k):

"""

Initialize your data structure here. Set the size of the deque to be k.

:type k: int

"""

self.\_size = 0

self.\_front, self.\_rear = 0, 0

self.\_capacity = k

self.\_data = [-1] \* k

def insertFront(self, value):

"""

Adds an item at the front of Deque. Return true if the operation is successful.

:type value: int

:rtype: bool

"""

if self.isFull():

return False

if self.isEmpty():

self.\_data[self.\_front] = value

else:

self.\_front = (self.\_front - 1) % self.\_capacity

self.\_data[self.\_front] = value

self.\_size += 1

return True

def insertLast(self, value):

"""

Adds an item at the rear of Deque. Return true if the operation is successful.

:type value: int

:rtype: bool

"""

if self.isFull():

return False

if self.isEmpty():

self.\_data[self.\_rear] = value

else:

self.\_rear = (self.\_rear + 1) % self.\_capacity

self.\_data[self.\_rear] = value

self.\_size += 1

return True

def deleteFront(self):

"""

Deletes an item from the front of Deque. Return true if the operation is successful.

:rtype: bool

"""

if self.isEmpty():

return False

self.\_data[self.\_front] = -1

self.\_front = (self.\_front + 1) % self.\_capacity

self.\_size -= 1

if self.isEmpty():

self.\_rear = self.\_front

return True

def deleteLast(self):

"""

Deletes an item from the rear of Deque. Return true if the operation is successful.

:rtype: bool

"""

if self.isEmpty():

return False

self.\_data[self.\_rear] = -1

self.\_rear = (self.\_rear - 1) % self.\_capacity

self.\_size -= 1

if self.isEmpty():

self.\_front = self.\_rear

return True

def getFront(self):

"""

Get the front item from the deque.

:rtype: int

"""

return self.\_data[self.\_front]

def getRear(self):

"""

Get the last item from the deque.

:rtype: int

"""

return self.\_data[self.\_rear]

def isEmpty(self):

"""

Checks whether the circular deque is empty or not.

:rtype: bool

"""

return self.\_size == 0

def isFull(self):

"""

Checks whether the circular deque is full or not.

:rtype: bool

"""

return self.\_size == self.\_capacity