Description

My implementation includes a CircularArrayQueue class and a main class for tests.
The constructor for the CircularArrayQueue initializes the queue array with a fixed capacity capacity and sets the front and rear indices to 0, as well as the size to 0.

This isEmpty method checks the size variable to see whether or not the queue is empty.
The isFull method compares the size and capacity variables to see whether or not the queue is full/hit maximum capacity.
The size method simply returns the size variable which is updated in the enqueue and dequeue methods so it reflects the size of the queue.
The enqueue method adds the given element to the rear of the queue. If the queue is full, it throws an exception. Otherwise, it updates the rear index by incrementing and using modular division, and assigns the element to that index in the queue array.
The dequeue method removes and returns the element at the front of the queue. If the queue is empty, it throws an exception. Otherwise, it retrieves the element at the front index of the queue array, updates the rear index by incrementing and using modular division, and returns the retrieved element.
The printQueue method prints the elements of the queue by iterating over the indices from front to rear (modulus the capacity) and printing the corresponding element in the queue array.

Complexity Analysis

The constructor initializes the array with a fixed capacity and has a time complexity of O(1).
isEmpty: O(1), this method checks if the queue is empty, which takes constant time.

isFull: O(1), this method checks if the queue is full, which takes constant time.

int size(): This method returns the size of the queue and takes constant time, i.e. O(1).

enqueue: O(1), this method adds an element to the rear of the queue. If the queue is full, it throws an exception, which takes constant time. Otherwise, it adds the element to the rear of the queue, which takes constant time.

dequeue: O(1), this method removes and returns the element at the front of the queue. If the queue is empty, it throws an exception, which takes constant time. Otherwise, it removes and returns the element at the front of the queue, which takes constant time.

printQueue: O(n), where n is the number of elements in the queue. The method iterates over each element in the queue and prints it, which takes linear time in the number of elements in the queue.