CS 0445 Spring 2022
Recitation Exercise 10

[Note: This exercise builds on Recitation Exercise 1. If you did not complete that exercise, you can get the solution from the course Canvas site. Use the Recitation Exercise 1 solution as a starting point for this exercise.

Introduction:

Recently in lecture we discussed iterators and the Iterator<T> interface. An example was discussed in which an iterator was added to the author's LList<T> class, resulting in the LinkedListWithIterator<T> class. Before completing this exercise, review Lecture 22 and carefully look over the code in the author's LinkedListWithIterator<T> class and also in the MyArrayIterable<T> class.

In Recitation Exercise 1 you implemented two primitive queues that satisfied the author's QueueInterface<T>. In this exercise you will take one of those implementations and add an iterator to it, so that a user will be able to iterate over all of the values of the queue.

Consider the following interface:

```
import java.util.*;
public interface QueueWithIteratorInterface<T>
{
  /** Adds a new entry to the back of this queue.
      @param newEntry  An object to be added. */
  public void enqueue(T newEntry);

  /** Removes and returns the entry at the front of this queue.
      @return  the object at the front of the queue or throw
     EmptyQueueException if the queue is empty. */
  public T dequeue();

  /**  Retrieves the entry at the front of this queue.
      @return  The object at the front of the queue or throw
     EmptyQueueException if the queue is empty */
  public T getFront();

  /** Detects whether this queue is empty.
      @return  True if the queue is empty, or false otherwise.*/
  public boolean isEmpty();

  /** Removes all entries from this queue. */
  public void clear();

 /*  Create and return a new Iterator<T> over the contents of
     this queue
  public Iterator<T> iterator();

} // end QueueWithIteratorInterface
```

This interface is the same interface that you saw in Recitation Exercise 1, with the addition of the iterator() method. You can obtain this interface in file QueueWithIteratorInterface.java.

In this exercise you will add the iterator() method to your PrimQ1<T> class to yield the PrimQ1WithIterator<T> class. PrimQ1WithIterator<T> will implement the QueueWithIteratorInterface<T>, as shown above. See Recitation Exercise 1 for more information and hints about the PrimQ1<T> class. If you were unable to complete Recitation Exercise 1, see the Canvas site for the course and use my solution as a starting point for this exercise. For hints on the iterator implementation, see the handout MyArrayIterable.java from the course handouts.

Once you have implemented your PrimQ1WithIterator<T> class, you can test it with the main program Rec10.java. The Rec10.java program should compile and run with your PrimQ1WithIterator<T> class, and the output should match that shown in file Rec10Out.txt. Read the program carefully so that you see how your class is being utilized in the program. Note that you will need the interface file QueueWithIteratorInterface.java and the exception file EmptyQueueException.java in order to compile your program, so be sure to download those files. Also note that for this exercise your queue should NOT implement the Moves interface.

Prior to the end of recitation, your TA will be soliciting volunteers to give a brief explanation and demonstration of their classes. As always, these are not required but I encourage everyone to consider doing this.