```
public class ArrayQueue<T> implements CarlQueue<T> {

   public T[] queue;
   private static final int DEFAULT_CAPACITY = 20;
   private int front;
   private int rear;

   public ArrayQueue() {
      queue = (T[])new Object[DEFAULT_CAPACITY];
      front = 0;
      rear = 0;
   }

   private boolean isFull() {
      return ((rear + 1) % queue.length == front);
   }

   public boolean isEmpty() {
      return (rear == front);
   }

   public T getFront() {
      if (!isEmpty()) {
         return queue[front];
      }
      else {
         return null;
      }
   }

   public void enqueue(T item) {
      if (isFull()) {
         // Create bigger array
         T[] biggerQueue = (T[])new Object[queue.length * 2];
         // Copy values into correct positions in new array
         for (int i=0; i < queue.length; i++) {

            biggerQueue[i] = queue[_____________________________________________];

         }
         rear = _____________________________________________;

         front = _____________________________________________;

         queue = biggerQueue;
      }

      // Add the new item to the queue
      queue[_____________________________________] = item;

      rear = _____________________________________________________;

   }


   public T dequeue() {
      if (!isEmpty()) {

         T itemToRemove = queue[_____________________________________________];

         front = _____________________________________________;

         return itemToRemove;
      }
      else {
         return null;
      }
   }


}
```