Queue and Stack Assignment

1. Create an instance of a LinkedList class as a Deque and behave as a FIFO queue:
   `Deque<String> queue = new LinkedList<String>();`
2. Create a loop that will accept input from the console. Prompt the user for a string in 1 of 3 formats (a verb and potentially a value):
   a. enqueue:value
   b. dequeue
   c. list
   d. quit
3. Using the String split() method, determine what was entered.
4. When the verb is "list", print to STDOUT the contents of the queue:
   `System.out.println(queue + "\n");`
5. When the verb is "dequeue" remove the head of the queue:
   `queue.removeFirst();Í`
6. When the verb is "enqueue" add the value to the rear of the queue (if a value was not entered, print a help message):
   `queue.addLast(value);`
7. When the verb is "quit" terminate the loop
8. When the verb is not one of these values, print a help message.
9. Refer to the Java docs for the Deque class

10. Create a second instance of a LinkedList class as a Deque and behave as a FIFO queue:
    `Deque<String> stack = new LinkedList<String>();`
11. Create a loop that will accept input from the console. Prompt the user for a string in 1 of 3 formats (a verb and potentially a value):
    a. push:value
    b. pop
    c. list
    d. quit
12. Using the String split() method, determine what was entered.
13. When the verb is "list", print to STDOUT the contents of the queue:
    `System.out.println(stack + "\n");`
14. When the verb is "pop" remove the head of the queue:
    `stack.removeFirst();`
15. When the verb is "push" add the value to the rear of the queue (if a value was not entered, print a help message):
    `stack.addFirst(value);`
16. When the verb is "quit" terminate the loop
17. When the verb is not one of these values, print a help message.