In lab this week, you will implement the Sequence ADT described on pages 145-159 of your textbook. It is important that you thoroughly understand the ADT. In addition to the methods listed in your textbook, you will include the two methods toString() and equals() as described below:

```
/**
     * Returns a String representation of this sequence.  If the sequence is
     * empty, the method should return <>.  If the sequence has one item,
     * say 1.1, and that item is not the current item, the method should
     * return <1.1>.  If the sequence has more than one item, they should
     * be separated by commas with a following space, for example:
     * <1.1, 2.2, 3.3>.  If there exists a current item, then that item
     * should be surrounded by square brackets.  For example, if the second
     * item is the current item, the method should return: <1.1, [2.2], 3.3>.
     *
     * @return a String representation of this sequence.
     */
    public String toString()

    /**
     * Determines if this object is equal to the other object.
     *
     * @return true if this object is equal to the other object, false
     *         otherwise
     */
    public boolean equals(Object other)
```

ASSIGNMENT: Look at the following program, StudentAssignment.java, which uses a DoubleArraySeq. The program creates a DoubleArraySeq and calls some of its methods. Fill in the blanks in the comments to show the values of the variables at the end of the statements (note that some of the if statements will have false conditions, and the variables inside will not change.) By the way, an empty sequence has no current element.

```
public class StudentAssignment {
    public static void main(String[] args) {
        DoubleArraySeq seq = new DoubleArraySeq();
        int answer = 1;
        int size = 0;
        double current = 0.0;
        String content = "";
        boolean isCurrent = false;
        int capacity;

        size = seq.size();                    //  1. size =  _______________

        capacity  = seq.getCapacity();        //  2. capacity =  ____________

        isCurrent = seq.isCurrent();          //  3. isCurrent =  ___________
        if (isCurrent)
           current = seq.getCurrent();        //  4. current = ______________
```

```
content = seq.toString();              //  5. content =  _____________
seq.trimToSize();
capacity = seq.getCapacity();          //  6. capacity =  _____________
seq.ensureCapacity(5);
capacity = seq.getCapacity();          //  7. capacity =  _____________
seq.addAfter(1.1);
content = seq.toString();              //  8. content =  _____________

size = seq.size();                     //  9. size = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 10. current = _____________
seq.addBefore(2.2);
content = seq.toString();              // 11. content =  _____________

size = seq.size();                     // 12. size =  _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 13. current = _____________
seq.addAfter(3.3);
content = seq.toString();              // 14. content = _____________

size = seq.size();                     // 15. size = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 16. current = _____________
seq.advance();
content = seq.toString();              // 17. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 18. current = _____________
seq.advance();
content = seq.toString();              // 19. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 20. current = _____________
seq.addBefore(4.4);
content = seq.toString();              // 21. content = _________________

size = seq.size();                     // 22. size = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 23. current = _____________
seq.advance();
content = seq.toString();              // 24. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 25. current = _______________
seq.advance();
content = seq.toString();              // 26. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 27. current = _____________
seq.advance();
content = seq.toString();              // 28. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 29. current = _____________
seq.advance();
content = seq.toString();              // 30. content = _________________
if (seq.isCurrent())
   current = seq.getCurrent();         // 31. current = _____________
```

```
        seq.start();
        content = seq.toString();              // 32. content = _________________
        if (seq.isCurrent())
           current = seq.getCurrent();         // 33. current = _________________
        seq.advance();
        content = seq.toString();              // 34. content = _________________
        seq.removeCurrent();
        content = seq.toString();              // 35. content = _________________

        size = seq.size();                     // 36. size = ___________________
        if (seq.isCurrent())
           current = seq.getCurrent();         // 37. current = _________________
        seq.removeCurrent();
        content = seq.toString();              // 38. content = _________________
        if (seq.isCurrent())
           current = seq.getCurrent();         // 39. current = _________________
        seq.removeCurrent();
        content = seq.toString();              // 40. content = _________________
        if (seq.isCurrent())
           current = seq.getCurrent();         // 41. current = _________________

        content = seq.toString();              // 42. content = _________________
        seq.start();
        content = seq.toString();              // 43. content = _________________
        seq.removeCurrent();
        content = seq.toString();              // 44. content = _________________
    }
}
```