Free Response Robot.docx

This question has three classes. You will implement two methods in the `FuelDepot` class.

A fuel depot has a number of fuel tanks arranged in a line and a robot that moves a filling mechanism back and forth along the line so that the tanks can be filled. A fuel tank is specified by the FuelTank method below.

```
public class FuelTank {
  private int level;

  public FuelTank(int level) { this.level = level; }
  public int getFuelLevel() { return level; }
}
```

A fuel depot keeps track of the fuel tanks and the robot. The following figure represents the tanks and the robot in a fuel depot. The robot, indicated by the arrow, is currently at index 2 and is facing to the right.

Tank index	0	1	2	3	4	5
Fuel Level in Tank	80	70	20	45	50	25

Robot 

```
/**
 * The state of the robot includes the index of its location and the direction in which
 * it is facing (to the right or to the left). This information is specified in the
 * FuelRobot interface as shown in the following declaration.
 */
public class FuelRobot
{
   private int currentIndex;
   private boolean facingRight;

   public FuelRobot(int p) {
      currentIndex = p;
   }

   /** @return the index of the current location of the robot */
   public int getCurrentIndex() {return currentIndex; }

   /** Determine whether the robot is currently facing to the right
    *  @return true if the robot is facing to the right (toward tanks with larger indexes)
    *         false if the robot is facing to the left (toward tanks with smaller indexes)
    */
   public boolean isFacingRight() { return facingRight; }
```

```
   /** Changes the current direction of the robot */
   public void changeDirection() { facingRight = !facingRight; }

   /** Moves the robot in its current direction by the number of locations specified.
    *  @param numLocs the number of locations to move. A value of 1 moves
    *              the robot to the next location in the current direction.
    *       Precondition: numLocs > 0
    */
   public void moveForward(int numLocs)
   {
      if (facingRight)
         currentIndex += numLocs;
      else
         currentIndex -= numLocs;
   }
}
```

Part (a) Write the *FuelDepot* method *nextTankToFill* that returns the index of the next tank to be filled. The index for the next tank to be filled is determined according to the following rules:

- Return the index of a tank with the lowest fuel level that is less than or equal to a given threshold.
- If there is more than one fuel tank with the same lowest fuel level, any of their indexes can be returned.
- If there are no tanks with a fuel level less than or equal to the threshold, return the robot's current index.

For example, suppose the tanks contain the fuel levels shown in the following figure.

Tank index	0	1	2	3	4	5	6
Fuel Level in Tank	20	30	80	55	50	75	20

The following table shows the results of several independent calls to nextTankToFill.

threshold	Return Value	Rationale
50	0 or 6	20 is the lowest fuel level, so either 0 or 6 can be returned.
15	2	There are no tanks with a fuel level <= threshold, so the robot's current index is returned.

Part (b) Write the `FuelDepot` method `moveToLocation` that will move the robot to the given tank location. Because the robot can only move forward, it may be necessary to change the direction of the robot before having it move. Do not move the robot past the end of the line of fuel tanks.

```
/**
 *  Expected output:
 *
 * Fuel Tank Levels: 20 30 80 55 50 75 35 70 15 25
 * Current position: 2
 * Is facing right: false
 * Threshold = 50, Next Tank to Fill = 8
 * Threshold = 10, Next Tank to Fill = 2
 * Moved to location 7
 * Is facing right: true
 * Moved to location 3
 * Is facing right: false
 * Moved to location 9
 * Is facing right: true
 * Moved to location 6
 * Is facing right: false
 * Moved to location 0
 * Is facing right: false
 */
import java.util.List;
import java.util.ArrayList;

public class FuelDepot
{
   /** The robot used to move the filling mechanism */
   private FuelRobot filler;

   /** The list of fuel tanks */
   private ArrayList<FuelTank> tanks;

   public FuelDepot(FuelRobot r, List<FuelTank> lst)
   {
      filler = r;
      tanks = new ArrayList<FuelTank>();
      for (FuelTank t : lst)
         tanks.add(t);
   }

   /** Determines and returns the index of the next tank to be filled.
    * @param threshold: fuel tanks with a fuel level <= threshold may be filled
    * @return index of the location of the next tank to be filled
    * Postcondition: the state of the robot has not changed
    */
   public int nextTankToFill(int threshold)
```

```
    { /** to be implemented in part (a) */
     }

    /** Moves the robot to location locIndex.
     * @param locIndex the index of the location of the tank to move to
     * Precondition: 0 <= locIndex < tanks.size()
     * Postcondition: the current location of the robot is locIndex
     */
    public void moveToLocation(int locIndex)
    { /** to be implemented in part (b) */
    }

    /*** The main method below is complete and test the code above ***/
    public static void main(String[] args)
    {
       int[] levels = {20, 30, 80, 55, 50, 75, 35, 70, 15, 25};
       ArrayList<FuelTank> tanks = new ArrayList<FuelTank>();
       System.out.print("Fuel Tank Levels: ");
       for (int level : levels) {
          tanks.add(new FuelTank(level));
          System.out.print(level + " ");
       }
       System.out.println("");
       FuelRobot filler = new FuelRobot(2);
       FuelDepot depot = new FuelDepot(filler, tanks);
       System.out.println("Current position: " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
       System.out.println("Threshold = 50, Next Tank to Fill = " +
          depot.nextTankToFill(50));
       System.out.println("Threshold = 10, Next Tank to Fill = " +
          depot.nextTankToFill(10));
       depot.moveToLocation(7);
       System.out.println("Moved to location " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
       depot.moveToLocation(3);
       System.out.println("Moved to location " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
       depot.moveToLocation(9);
       System.out.println("Moved to location " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
       depot.moveToLocation(6);
       System.out.println("Moved to location " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
       depot.moveToLocation(0);
       System.out.println("Moved to location " + filler.getCurrentIndex());
       System.out.println("Is facing right: " + filler.isFacingRight());
    }
}
```