ArrayList FuelTank FRQ

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* class below.

```
class FuelTank
{
  // @return an integer value that ranges from0 (empty) to 100 (full)
  public int getFuelLevel()
  { return level; }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

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* class as shown in the following declaration.

```
class FuelRobot
{
  // @return the index of the current location of the robot
  public int getCurrentIndex()
  { // Implementation not shown. }

  /** 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()
  { // Implementation not shown. }

  // Changes the current direction of the robot
  public void changeDirection()
  { // Implementation not shown. }

  /** 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)
  { // Implementation not shown. }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

A fuel depot is represented by the *FuelDepot* class as shown in the following class declaration.

```java
public class FuelDepot {

    // The robot used to move the filling mechanism.
    private FuelRobot filler;

    // The list of fuel tanks.
    private List<FuelTank> tanks;

    /** 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) */  }
    // There may be instance variables, constructors, and methods that are not
    shown.
}
```

(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
Robot			->				

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.

Complete the method `nextTankToFill()` below.

```
/** 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)
```

(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 <u>not</u> move the robot past the end of the line of fuel tanks.

Complete the method `moveToLocation ()` below.

```
/** 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)
```