```java
package project3;

import org.junit.Before;
import org.junit.Test;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.GregorianCalendar;

import static org.junit.Assert.*;

public class ListEngineTest {

    SimpleDateFormat df;
    GregorianCalendar date1;
    GregorianCalendar date2;
    ListEngine DList;

    @Before
    public void before() {
        df = new SimpleDateFormat("MM/dd/yyyy");
        date1 = new GregorianCalendar();
        date2 = new GregorianCalendar();
        try {
            Date d1 = df.parse("3/20/2019");
            date1.setTime(d1);
            Date d2 = df.parse("9/20/2019");
            date2.setTime(d2);
        } catch (Exception e) {
            //This is needed in order to parse something in case it fails
        }
        DList = new ListEngine();

        //sell a car
        Date d;
        try{
            d = df.parse("1/2/2030");
        } catch(Exception e){
            return;
        }
        GregorianCalendar temp = new GregorianCalendar();
        temp.setTime(d);
        Auto auto = DList.get(0);
        auto.setNameOfBuyer("John");
        auto.setSoldOn(temp);
        auto.setSoldPrice(1000);

    }

    @Test
    public void getColumnNameBought() {
        String col = DList.getColumnName(5);

        assertEquals("Turbo", col);
    }

    @Test
    public void getColumnNameSold() {
        DList.updateDisplay(DList.soldScreen);
        String col = DList.getColumnName(5);

        assertEquals("Sold On", col);
    }

    @Test
    public void getColumnNameOverdue() {
        DList.updateDisplay(DList.overdueScreen);
        String col = DList.getColumnName(3);

        assertEquals("Days Overdue", col);
    }

    @Test (expected = RuntimeException.class)
    public void getColumnNameOutOfRange() {
        String col = DList.getColumnName(10);
    }

    @Test
    public void updateDisplay() {
        DList.updateDisplay(DList.soldScreen);

        assertEquals(DList.soldScreen, DList.displayValue);
    }

    @Test
    public void add() {
        Car Car1 = new Car(date1, "Outback", "Buyer1", "LX", false);
        Car Car2 = new Car(date2, "Chevy", "Buyer2", "EX", false);
        DList.add(Car1);
        DList.add(Car2);

        // 8 total but only 7 in bought screen
        assertEquals(7, DList.getSize());
    }

    @Test
    public void get() {
        Auto first = DList.get(0);

        assertEquals(first.autoName, "F350");
    }

    @Test
    public void getSize() {
        int size = DList.getSize();

        assertEquals(6, size);
    }

    @Test
    public void getRowCount() {
        int numRows = DList.getRowCount();

        assertEquals(6, numRows);
    }

    @Test
    public void getColumnCountBought() {
        int numColumns = DList.getColumnCount();

        assertEquals(6, numColumns);
    }

    @Test
    public void getColumnCountSold() {
        DList.updateDisplay(DList.soldScreen);
        int numColumns = DList.getColumnCount();

        assertEquals(6, numColumns);
    }

    @Test
    public void getColumnCountOverdue() {
        DList.updateDisplay(DList.overdueScreen);
        int numColumns = DList.getColumnCount();

        assertEquals(4, numColumns);
    }


    // cols 0-2 any screen
    // col 3 all 3
    // col 4 & 5 bought sold

    // bought - col 0, 1, 2, 3, 4, 5
    // sold - 3, 4, 5
    // overdue - 3
    // out of range
    @Test
    public void getValueAtBought0() {
        String val = DList.getValueAt(0, 0).toString();

        assertEquals(val, "F350");
    }

    @Test
    public void getValueAtBought1() {
        String val = DList.getValueAt(0, 1).toString();

        assertEquals(val, "0.0");
    }

    @Test
    public void getValueAtBought2() {
        String val = DList.getValueAt(0, 2).toString();

        assertEquals(val, "01/20/2010");
    }

    @Test
    public void getValueAtBought3() {
        String val = DList.getValueAt(0, 3).toString();

        assertEquals(val, "EX");
    }

    @Test
    public void getValueAtBought4Truck() {
        String val = DList.getValueAt(0, 4).toString();

        assertEquals(val, "true");
    }

    @Test
    public void getValueAtBought4Car() {
        String val = DList.getValueAt(1, 4).toString();

        assertEquals(val, "");
    }

    @Test
    public void getValueAtBought5Truck() {
        String val = DList.getValueAt(0, 5).toString();

        assertEquals(val, "");
    }

    @Test
    public void getValueAtBought5Car() {
        String val = DList.getValueAt(1, 5).toString();

        assertEquals(val, "false");
    }

    @Test
    public void getValueAtSold3() {
        DList.updateDisplay(DList.soldScreen);
        String val = DList.getValueAt(0, 3).toString();

        assertEquals(val, "John");
    }

    @Test
    public void getValueAtSold4() {
        DList.updateDisplay(DList.soldScreen);
        String val = DList.getValueAt(0, 4).toString();

        assertEquals(val, "1000.0");
    }

    @Test
    public void getValueAtSold5() {
        DList.updateDisplay(DList.soldScreen);
        String val = DList.getValueAt(0, 5).toString();

        assertEquals(val, "01/02/2030");
    }

    @Test
    public void getValueAtOverdue3() {
        DList.updateDisplay(DList.overdueScreen);
        String val = DList.getValueAt(0, 3).toString();

        assertEquals(val, "316");
    }

    @Test (expected = RuntimeException.class)
    public void getValueAtOutOfRange() {
        String val = DList.getValueAt(0, 10).toString();
    }

    @Test
    public void saveDatabase() {
        DList.saveDatabase("JUnit test");
    }

    @Test
    public void loadDatabase() {
        DList.loadDatabase("JUnit test");
        String val = DList.get(0).getAutoName();

        assertEquals("F350", val);
    }

    @Test
    public void saveAsTextError() {
        boolean b = DList.saveAsText("");

        assertFalse(b);
    }

    @Test
    public void saveAsText() {
        DList.saveAsText("JUnit text test");
    }

    @Test
    public void loadFromText() {
        DList.loadFromText("JUnit text test");
        String val = DList.get(0).getAutoName();

        assertEquals("Outback", val);
    }
}

```