

ASSIGNMENT 4

NUID:002644926

1) SINGLETON PATTERN

Sample run:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and the current project name 'SingletonDesignPattern - Apache NetBeans IDE...'. The toolbar has various icons for file operations. The left sidebar shows the 'Projects' view with several Java projects listed, and the 'Output' view showing build logs. The main workspace displays the code for 'SingleObject.java' under the package 'SingletonPatternDemo'. The code implements the Singleton pattern with a static factory method 'getInstance()' and a private constructor. The output window shows a successful build with the message 'Hello World!'.

```
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4
5  /**
6   *
7   * @author Namitha
8   * @version 1.0
9   */
10  public class SingletonPatternDemo {
11
12      /**
13      * Main method to test the singleton design pattern.
14      *
15      * @param args command line arguments
16      */
17      public static void main(String[] args) {
18          SingleObject object = singleObject.getInstance();
19          object.showMessage();
20      }
21  }
```

```
run:
Hello World!
BUILD SUCCESSFUL (total time: 0 seconds)
```

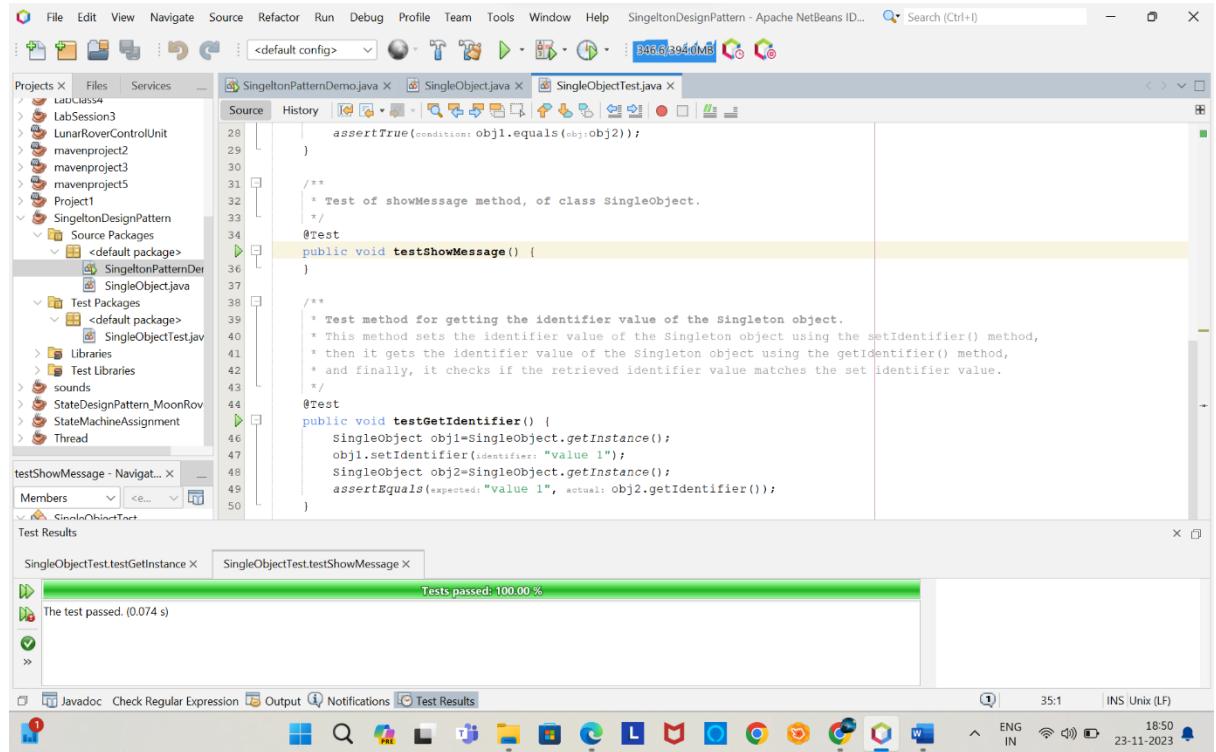
Test case 1:

The screenshot shows the Apache NetBeans IDE interface with the 'FacadeDesignPattern - Apache NetBeans IDE 19' project selected. The main workspace displays the code for 'ShapeMakerTest.java' under the package 'FacadeDesignPattern'. The code contains two test methods: 'testDrawCircle()' and 'testDrawRectangle()'. The 'testDrawCircle()' method creates a Circle object and compares its draw method output with the ShapeMaker's drawCircle method using assertEquals. The 'testDrawRectangle()' method creates a Rectangle object and compares its draw method output with the ShapeMaker's drawRectangle method using assertEquals. The bottom pane shows the 'Test Results' window with a green bar indicating 'Tests passed: 100.00 %' and the message 'The test passed. (0.073 s)'.

```
15
16      ShapeMaker shapeMaker = new ShapeMaker();
17
18      /**
19       * Test of drawCircle method, of class ShapeMaker.
20       */
21      @Test
22      public void testDrawCircle() {
23          // Create an instance of Circle
24          Circle c1 = new Circle();
25
26          // Compare the output of Circle's draw method with ShapeMaker's drawCircle method
27          assertEquals(expected:c1.draw(), actual: shapeMaker.drawCircle());
28
29      /**
30       * Test of drawRectangle method, of class ShapeMaker.
31       */
32      @Test
33      public void testDrawRectangle() {
34          // Create an instance of Rectangle
35          Rectangle r1 = new Rectangle();
36
37          // Compare the output of Rectangle's draw method with ShapeMaker's drawRectangle method
38      }
```

Tests passed: 100.00 %
The test passed. (0.073 s)

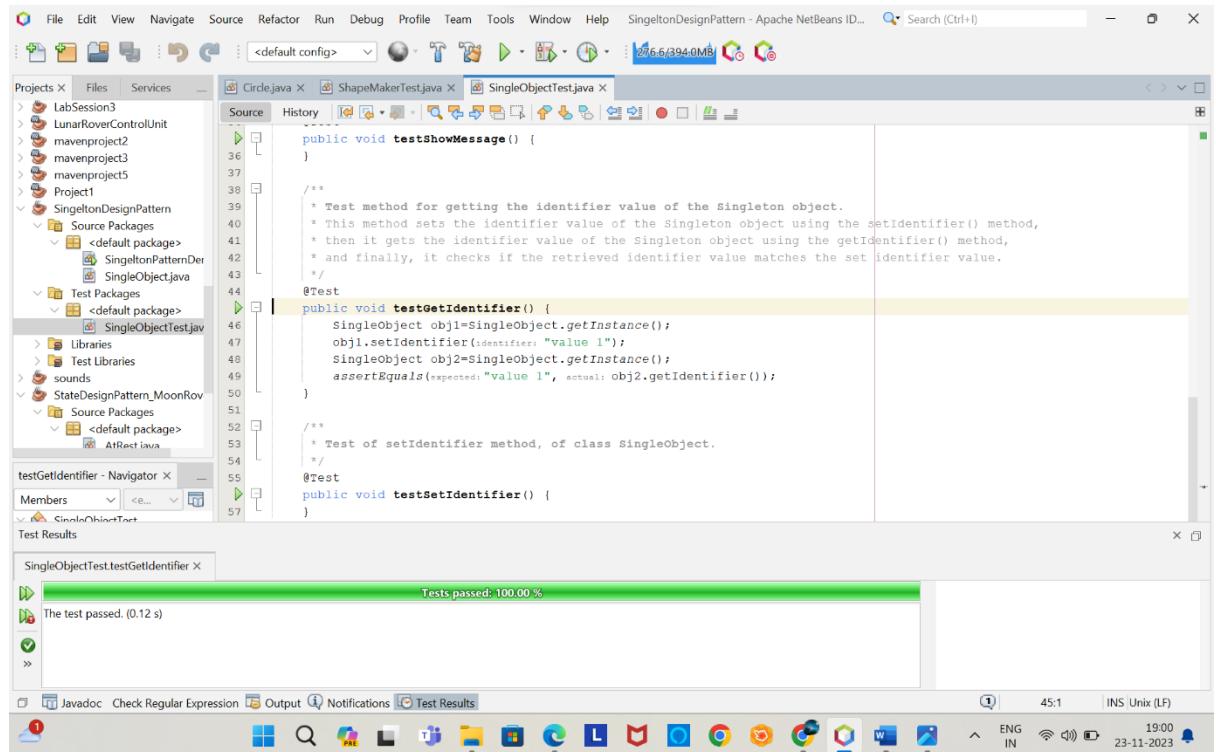
Test case 2:



The screenshot shows the NetBeans IDE interface with the following details:

- Projects X**: Shows various projects like LabSession3, LunarRoverControlUnit, mavenproject2, mavenproject3, mavenproject5, Project1, and SingletonDesignPattern.
- SingleObjectTest.java**: The active file contains Java code for testing the SingleObject class. It includes two test methods: `testShowMessage()` and `testGetIdentifier()`. The `testShowMessage()` method asserts that two objects are equal. The `testGetIdentifier()` method sets an identifier, gets it again, and asserts they are equal.
- Test Results**: A summary window shows "Tests passed: 100.00 %". Below it, a detailed view for `SingleObjectTest.testShowMessage` shows "The test passed. (0.074 s)".
- Toolbar**: Standard NetBeans toolbar with icons for file operations, search, and project navigation.
- System Tray**: Shows system status including battery level, signal strength, and date/time (23-11-2023).

Test case 3:



The screenshot shows the NetBeans IDE interface with the following details:

- Projects X**: Shows various projects like LabSession3, LunarRoverControlUnit, mavenproject2, mavenproject3, mavenproject5, Project1, and SingletonDesignPattern.
- SingleObjectTest.java**: The active file contains Java code for testing the SingleObject class. It includes three test methods: `testShowMessage()`, `testGetIdentifier()`, and `testSetIdentifier()`. The `testGetIdentifier()` method checks if two objects have the same identifier after setting it. The `testSetIdentifier()` method checks if the identifier can be set and retrieved correctly.
- Test Results**: A summary window shows "Tests passed: 100.00 %". Below it, a detailed view for `SingleObjectTest.testGetIdentifier` shows "The test passed. (0.12 s)".
- Toolbar**: Standard NetBeans toolbar with icons for file operations, search, and project navigation.
- System Tray**: Shows system status including battery level, signal strength, and date/time (23-11-2023).

Test case 4:

```
38 /**
39  * Test method for getting the identifier value of the Singleton object.
40  * This method sets the identifier value of the Singleton object using the setIdentifier() method,
41  * then it gets the identifier value of the Singleton object using the getIdentifier() method,
42  * and finally, it checks if the retrieved identifier value matches the set identifier value.
43 */
44 @Test
45 public void testGetIdentifier() {
46     SingleObject obj1=SingleObject.getInstance();
47     obj1.setIdentifier(identifier: "value 1");
48     SingleObject obj2=SingleObject.getInstance();
49     assertEquals(expected: "value 1", actual: obj2.getIdentifier());
50 }
51 /**
52  * Test of setIdentifier method, of class SingleObject.
53 */
54 @Test
55 public void testSetIdentifier() {
56 }
57 }
58 }
59 }
```

The test passed. (0.088 s)

All Test cases:

```
1 /**
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/UnitTests/JUnit4TestClass.java to edit this template
4 */
5
6 import org.junit.Test;
7 import static org.junit.Assert.*;
8
9 /**
10  * @author Namitha
11  * @version 1.0
12 */
13
14 public class SingleObjectTest {
15
16     public SingleObjectTest() {
17     }
18
19 /**
20  * Test method for checking if the getInstance() method always returns the same instance of the SingleObject class.
21  * This method gets two instances of the SingleObject class using the getInstance() method,
22  * and finally, it checks if both instances are equal, meaning that they refer to the same object in memory.
23 */
24 }
```

All 4 tests passed. (0.156 s)

2) FAÇADE DESIGN PATTERN

Sample run:

The screenshot shows the Apache NetBeans IDE interface. The Projects tab displays a hierarchy of Java packages: College, FacadeDesignPattern, Source Packages, and Test Packages. Under Source Packages, there are files like Circle.java, FacadePatternDemo.java, Rectangle.java, Shape.java, ShapeMaker.java, and Square.java. Under Test Packages, there is ShapeMakerTest.java. The Editor tab shows the code for FacadePatternDemo.java, which contains a main method that creates a ShapeMaker object and calls its drawCircle, drawRectangle, and drawSquare methods. The Output tab shows a successful build message: "BUILD SUCCESSFUL (total time: 0 seconds)". The bottom status bar indicates the date and time as 23-11-2023.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/classes/Class.java to edit this template
 */

/**
 * Example of Facade design pattern.
 *
 * @author Namitha
 * @version 1.0
 */
public class FacadePatternDemo {
    public static void main(String[] args) {
        ShapeMaker shapeMaker= new ShapeMaker();

        shapeMaker.drawCircle();
        shapeMaker.drawRectangle();
        shapeMaker.drawSquare();
    }
}
```

Test case 1:

The screenshot shows the Apache NetBeans IDE interface. The Projects tab displays a hierarchy of Java packages: College, FacadeDesignPattern, Source Packages, and Test Packages. Under Test Packages, there is ShapeMakerTest.java. The Editor tab shows the code for ShapeMakerTest.java, which contains two test methods: testDrawCircle and testDrawRectangle. Both tests create instances of Circle and Rectangle respectively, and compare their draw methods with ShapeMaker's drawCircle and drawRectangle methods using assertEquals. The Test Results tab shows a green bar indicating "Tests passed: 100.00 %". The bottom status bar indicates the date and time as 23-11-2023.

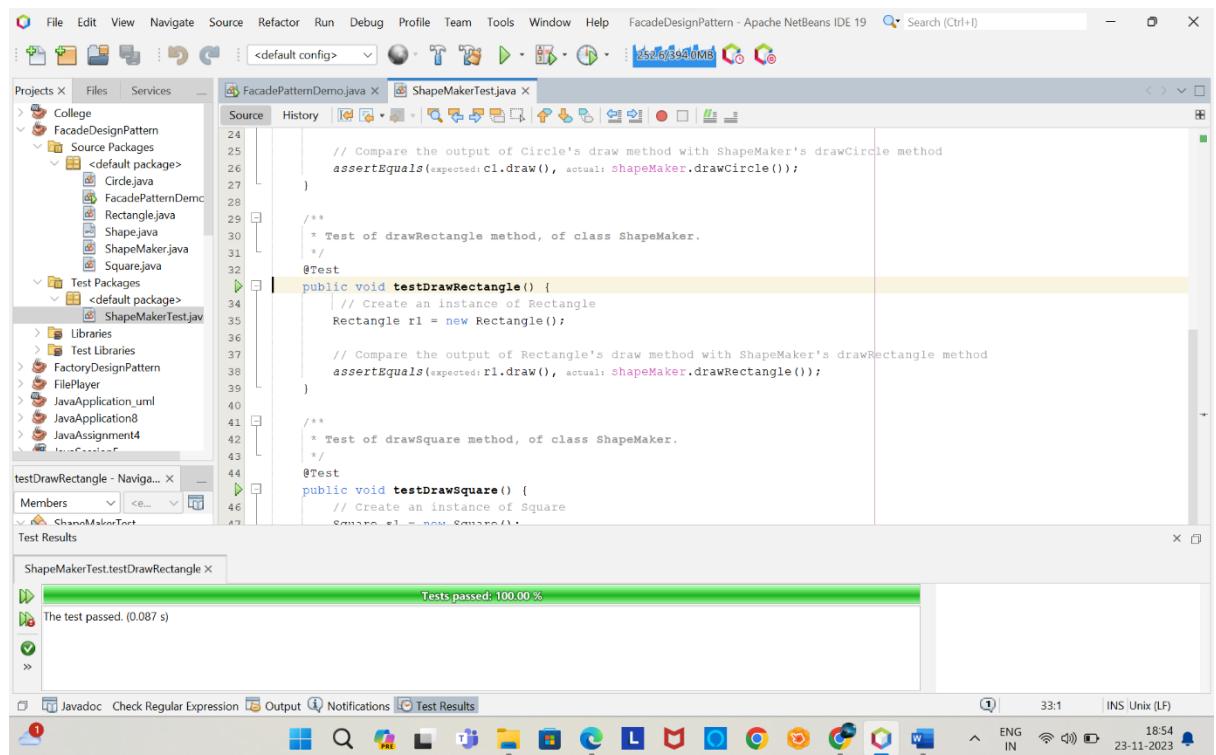
```
/*
 * Test of drawCircle method, of class ShapeMaker.
 */
@Test
public void testDrawCircle() {
    // Create an instance of Circle
    Circle c1 = new Circle();

    // Compare the output of Circle's draw method with ShapeMaker's drawCircle method
    assertEquals(expected:c1.draw(), actual: shapeMaker.drawCircle());
}

/*
 * Test of drawRectangle method, of class ShapeMaker.
 */
@Test
public void testDrawRectangle() {
    // Create an instance of Rectangle
    Rectangle r1 = new Rectangle();

    // Compare the output of Rectangle's draw method with ShapeMaker's drawRectangle method
}
```

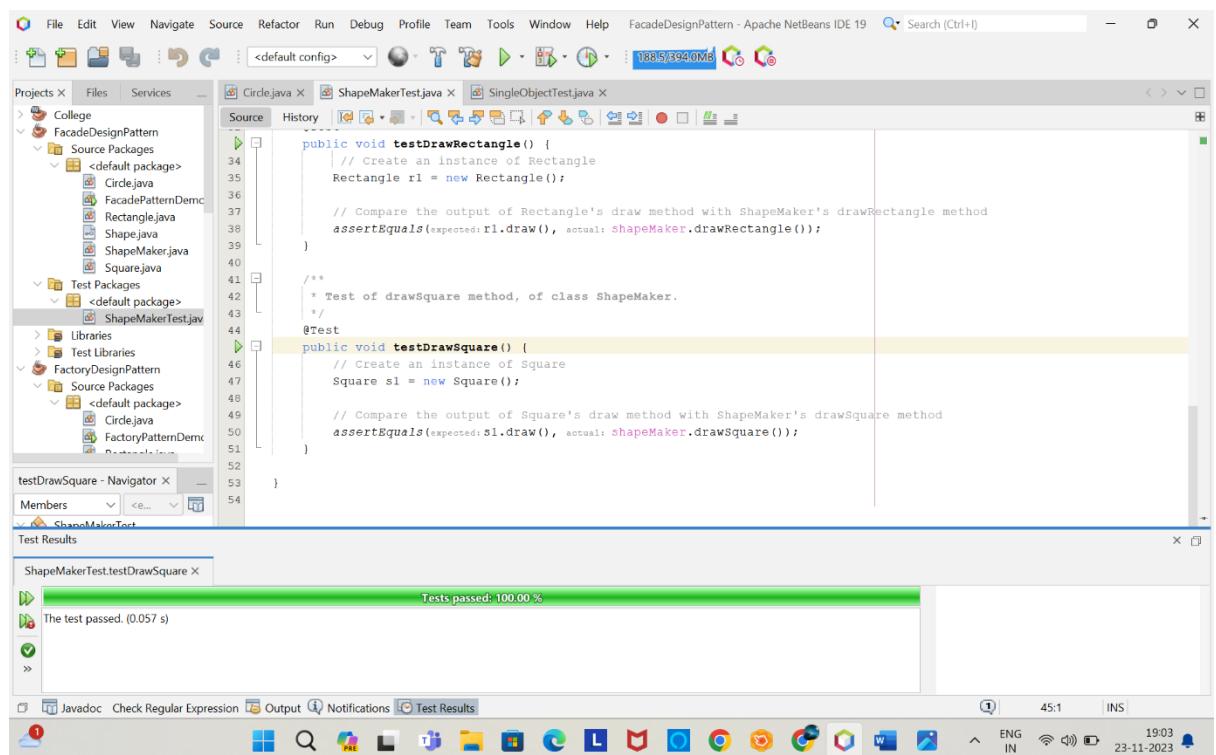
Test case 2:



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Menu Bar:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Standard NetBeans toolbar with icons for file operations, search, and project navigation.
- Project Explorer:** Shows a project named "FacadeDesignPattern" with packages "Source Packages" and "Test Packages".
- Code Editor:** The active file is "ShapeMakerTest.java" under "Test Packages". It contains Java test code for the "ShapeMaker" class, specifically for the "draw" method of "Shape" objects.
- Output:** Shows "2526/3940MB" of memory usage.
- Test Results:** A "Test Results" window shows "ShapeMakerTest.testDrawRectangle" with a green bar indicating "Tests passed: 100.00 %". Below it, a message says "The test passed. (0.087 s)".
- System Tray:** Shows system icons for battery, signal strength, and date/time (23-11-2023).

Test case 3:



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Menu Bar:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Standard NetBeans toolbar with icons for file operations, search, and project navigation.
- Project Explorer:** Shows a project named "FactoryDesignPattern" with packages "Source Packages" and "Test Packages".
- Code Editor:** The active file is "ShapeMakerTest.java" under "Test Packages". It contains Java test code for the "ShapeMaker" class, specifically for the "draw" method of "Shape" objects.
- Output:** Shows "1885/3940MB" of memory usage.
- Test Results:** A "Test Results" window shows "ShapeMakerTest.testDrawSquare" with a green bar indicating "Tests passed: 100.00 %". Below it, a message says "The test passed. (0.057 s)".
- System Tray:** Shows system icons for battery, signal strength, and date/time (23-11-2023).

All Test cases:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and a search bar. The title bar indicates "FacadeDesignPattern - Apache NetBeans IDE 19".

The left sidebar displays the project structure under "Projects X". It includes a "Source Packages" section with Circle.java, FacadePatternDemo.java, Rectangle.java, Shape.java, ShapeMaker.java, and Square.java. A "Test Packages" section contains ShapeMakerTest.java. There are also "Libraries" and "Test Libraries" sections.

The main editor window shows the code for ShapeMakerTest.java:

```
1 /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/UnitTests/JUnit4TestClass.java to edit this template
4  */
5
6 import org.junit.Test;
7 import static org.junit.Assert.*;
8 /**
9  * @author Namitha
10 * @version 1.0
11 */
12 public class ShapeMakerTest {
13
14     public ShapeMakerTest() {
15     }
16     ShapeMaker shapeMaker = new ShapeMaker();
17     /**
18      * Test of drawCircle method, of class ShapeMaker.
19     */
20     @Test
21     public void testDrawCircle() {
22         // Create an instance of circle
23         Circle cl = new Circle();
```

The "Test Results" panel at the bottom shows the outcome of the test run:

- ShapeMakerTest
- Tests passed: 100.00 %
- All 3 tests passed. (0.089 s)

The status bar at the bottom right shows system information: ENG IN, 18:53, 23-11-2023.

3) FACTORY DESIGN PATTERN

Sample run:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and a search bar. The title bar indicates "FactoryDesignPattern - Apache NetBeans IDE 19".

The left sidebar displays the project structure under "Projects X". It includes a "Source Packages" section with Circle.java, FacadePatternDemo.java, Rectangle.java, Shape.java, ShapeMaker.java, and Square.java. A "Test Packages" section contains ShapeMakerTest.java. There are also "Libraries" and "Test Libraries" sections.

The main editor window shows the code for Circle.java:

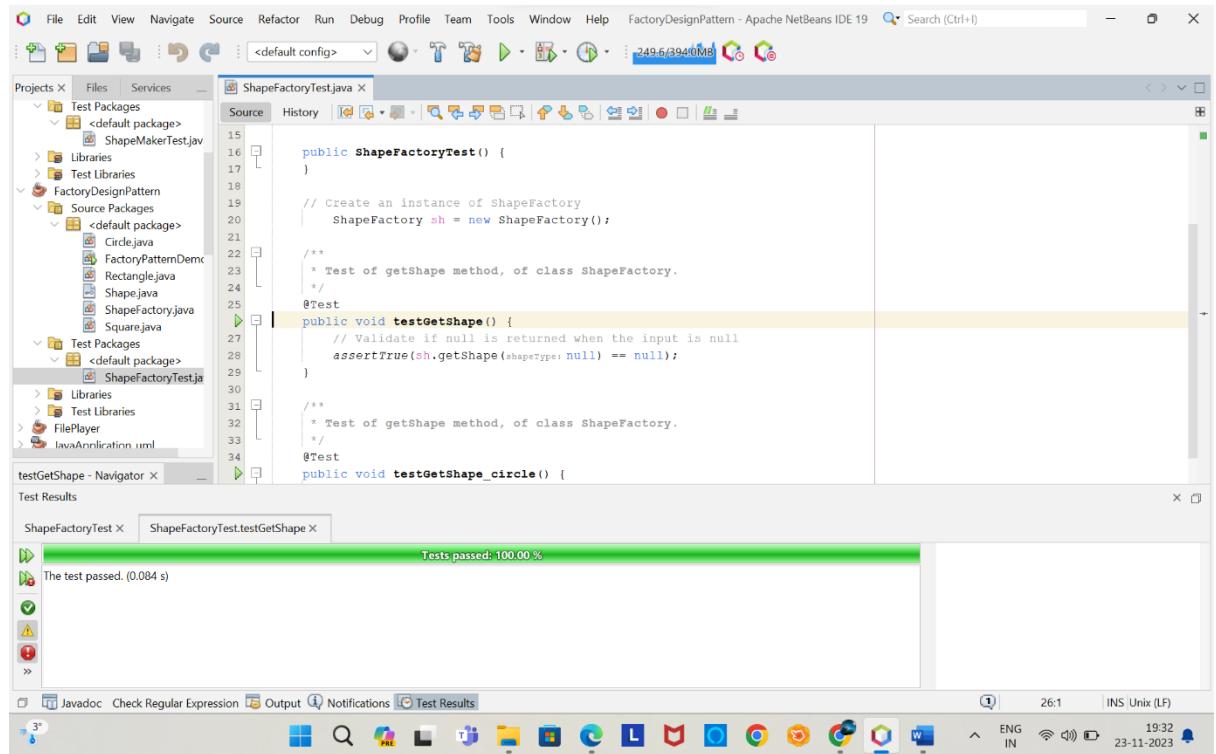
```
1 /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5
6 /**
7  * Circle shape performing the Shape interface.
8  *
9  * @author Namitha
10 * @version 1.0
11 */
12 public class Circle implements Shape{
13
14     @Override
15     public void draw(){
16         System.out.println("Circle:draw()");
17     }
18 }
```

The "Output" panel at the bottom shows the sample run output:

```
run:
Circle:draw()
Rectangle:draw()
Square:draw()
BUILD SUCCESSFUL (total time: 0 seconds)
```

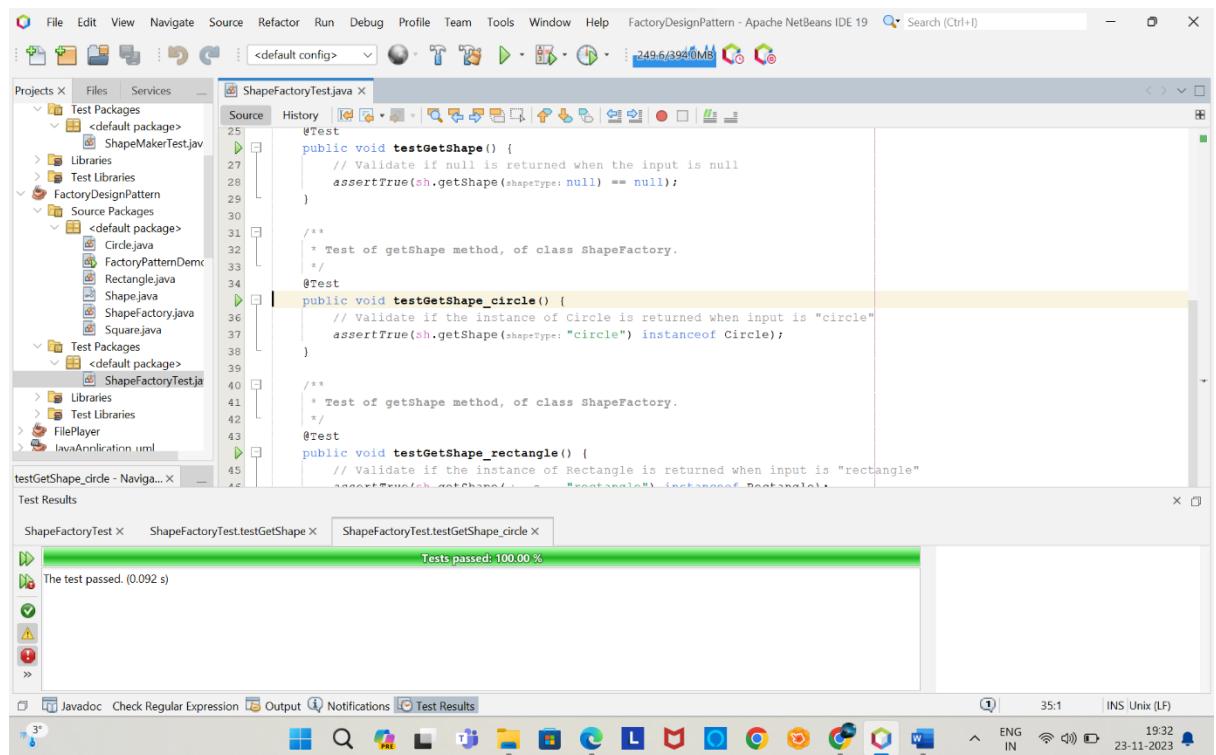
The status bar at the bottom right shows system information: ENG IN, 18:56, 23-11-2023.

Test case 1:



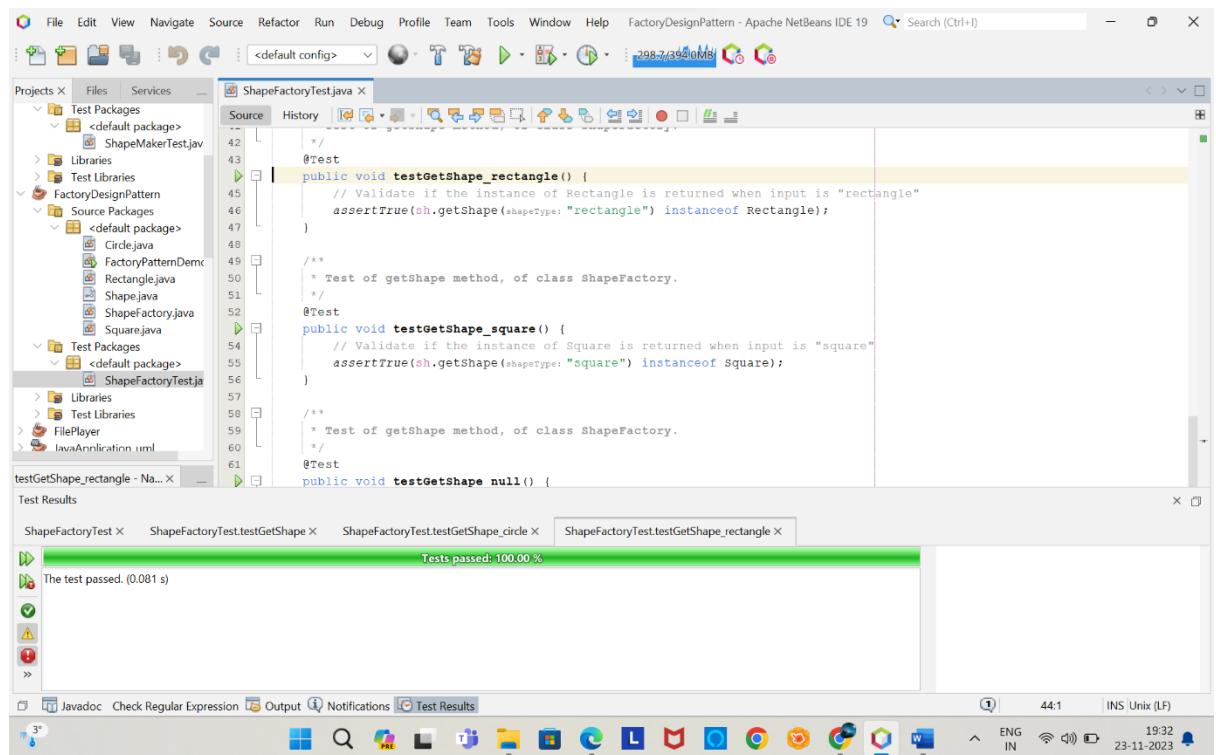
The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and a search bar. The title bar indicates "FactoryDesignPattern - Apache NetBeans IDE 19". The left sidebar displays the project structure under "Projects X" with packages like Test Packages, <default package>, Libraries, Test Libraries, FactoryDesignPattern, and Source Packages. The main editor window shows the code for "ShapeFactoryTest.java". The code contains several test methods: `testGetShape()`, `testGetShape_circle()`, and `testGetShape_rectangle()`. The code uses annotations like `@Test` and assertions like `assertTrue(sh.getShape(shapeType) == null)`. The bottom status bar shows "Tests passed: 100.00 %".

Test case 2:



This screenshot is identical to the one above, showing the Apache NetBeans IDE interface with the same project structure and code for "ShapeFactoryTest.java". The main difference is the test method highlighted in the code editor, which is `testGetShape_circle()`. The status bar at the bottom again shows "Tests passed: 100.00 %".

Test case 3:



The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Search (Ctrl+I). The title bar indicates "FactoryDesignPattern - Apache NetBeans IDE 19". The bottom status bar shows memory usage (298.7/394.0MB), network connection (WIFI), battery level (19:32), and system information (ENG IN, 23-11-2023).

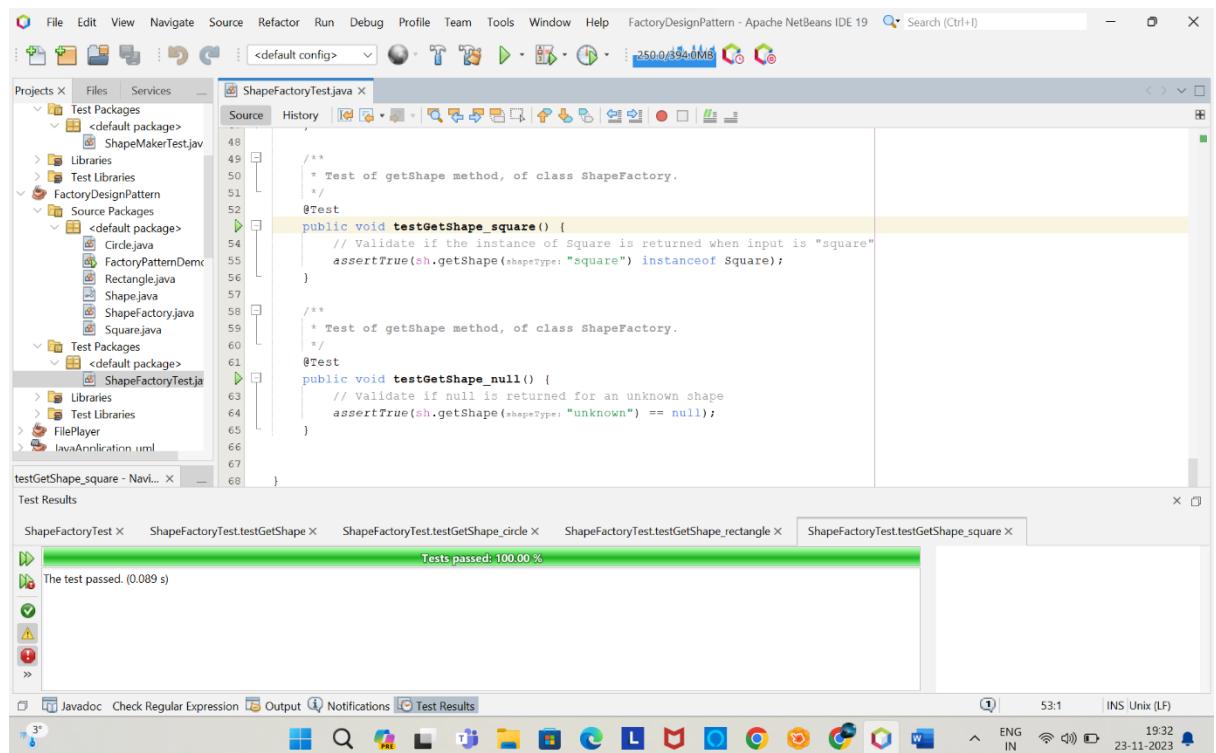
The left sidebar displays the project structure under "Projects X". It includes "Test Packages" (ShapeMakerTest.java), "Libraries", "Test Libraries", "FactoryDesignPattern" (Source Packages: Circle.java, FactoryPatternDemo.java, Rectangle.java, Shape.java, Shapefactory.java, Square.java), and "Test Packages" (ShapefactoryTest.java).

The main editor window shows the code for "ShapeFactoryTest.java". The code contains three test methods: `testGetShape_rectangle()`, `testGetShape_square()`, and `testGetShape_null()`. The `testGetShape_square()` method is highlighted.

```
42     */
43     @Test
44     public void testGetShape_rectangle() {
45         // Validate if the instance of Rectangle is returned when input is "rectangle"
46         assertTrue(sh.getShape(shapeType: "rectangle") instanceof Rectangle);
47     }
48
49     /**
50      * Test of getShape method, of class ShapeFactory.
51     */
52     @Test
53     public void testGetShape_square() {
54         // Validate if the instance of Square is returned when input is "square"
55         assertTrue(sh.getShape(shapeType: "square") instanceof Square);
56     }
57
58     /**
59      * Test of getShape method, of class ShapeFactory.
60     */
61     @Test
62     public void testGetShape_null() {
63         // Validate if null is returned for an unknown shape
64         assertEquals(sh.getShape(shapeType: "unknown"), null);
65     }
66 }
67
68 }
```

The "Test Results" tab at the bottom shows the status bar with "Tests passed: 100.00 %".

Test case 4:



The screenshot shows the Apache NetBeans IDE interface, identical to the previous one but with a different set of test results.

The top menu bar, title bar, and bottom status bar are the same as in the previous screenshot.

The left sidebar shows the same project structure: "Test Packages" (ShapeMakerTest.java), "Libraries", "Test Libraries", "FactoryDesignPattern" (Source Packages: Circle.java, FactoryPatternDemo.java, Rectangle.java, Shape.java, Shapefactory.java, Square.java), and "Test Packages" (ShapefactoryTest.java).

The main editor window shows the same code for "ShapeFactoryTest.java" as in the previous screenshot, with the `testGetShape_square()` method highlighted.

```
42     */
43     @Test
44     public void testGetShape_rectangle() {
45         // Validate if the instance of Rectangle is returned when input is "rectangle"
46         assertTrue(sh.getShape(shapeType: "rectangle") instanceof Rectangle);
47     }
48
49     /**
50      * Test of getShape method, of class ShapeFactory.
51     */
52     @Test
53     public void testGetShape_square() {
54         // Validate if the instance of Square is returned when input is "square"
55         assertTrue(sh.getShape(shapeType: "square") instanceof Square);
56     }
57
58     /**
59      * Test of getShape method, of class ShapeFactory.
60     */
61     @Test
62     public void testGetShape_null() {
63         // Validate if null is returned for an unknown shape
64         assertEquals(sh.getShape(shapeType: "unknown"), null);
65     }
66 }
67
68 }
```

The "Test Results" tab at the bottom shows the status bar with "Tests passed: 100.00 %".

Test case 5:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and a search bar. The title bar indicates "FactoryDesignPattern - Apache NetBeans IDE 19". The left sidebar displays the project structure under "Projects X" with packages like "Test Packages", "Libraries", "FactoryDesignPattern", and "Source Packages". The main editor window shows the code for "ShapeFactoryTest.java". The code contains two test methods: `testGetShape_square()` and `testGetShape_null()`. The `testGetShape_null()` method is highlighted with a yellow background. The status bar at the bottom right shows "352/3940MB" and the date "23-11-2023".

```
48
49     /**
50      * Test of getShape method, of class ShapeFactory.
51     */
52     @Test
53     public void testGetShape_square() {
54         // Validate if the instance of Square is returned when input is "square"
55         assertTrue(sh.getShape(shapeType: "square") instanceof Square);
56     }
57
58     /**
59      * Test of getShape method, of class ShapeFactory.
60     */
61     @Test
62     public void testGetShape_null() {
63         // Validate if null is returned for an unknown shape
64         assertTrue(sh.getShape(shapeType: "unknown") == null);
65     }
66
67 }
68 }
```

The "Test Results" panel at the bottom shows a green bar indicating "Tests passed: 100.00 %". Below it, a message says "The test passed. (0.089 s)". The status bar at the bottom right shows "62:1" and "INS Unix (LF)".

All Test cases:

The screenshot shows the Apache NetBeans IDE interface, similar to the previous one but with more test cases. The main editor window shows the code for "ShapeFactoryTest.java". It includes three test methods: `testGetShape()`, `testGetShape_circle()`, and `testGetShape_square()`. The `testGetShape_square()` method is highlighted with a yellow background. The status bar at the bottom right shows "218/3940MB" and the date "23-11-2023".

```
18
19     // Create an instance of ShapeFactory
20     ShapeFactory sh = new ShapeFactory();
21
22     /**
23      * Test of getShape method, of class ShapeFactory.
24     */
25     @Test
26     public void testGetShape() {
27         // Validate if null is returned when the input is null
28         assertTrue(sh.getShape(shapeType: null) == null);
29     }
30
31     /**
32      * Test of getShape method, of class ShapeFactory.
33     */
34     @Test
35     public void testGetShape_circle() {
36         // Validate if the instance of Circle is returned when input is "circle"
37         assertTrue(sh.getShape(shapeType: "circle") instanceof Circle);
38     }
39 }
```

The "Test Results" panel at the bottom shows a green bar indicating "Tests passed: 100.00 %". Below it, a message says "All 5 tests passed. (0.163 s)". The status bar at the bottom right shows "1:1" and "INS".

4) STATE DESIGN PATTERN-MOON ROVER

Sample run:

The screenshot shows the Apache NetBeans IDE interface. The Projects tab displays several Java files under the 'StateDesignPattern_MoonRover' project. The Output tab shows the console logs from the application's execution:

```
Current SubState: Constant Speed
Transitioning from Constant Speed State to Decelerate State...
Current State: Move Backward
Current SubState: Decelerate
Transitioning from Decelerate State to Accelerate State...
Current State: Move Backward
Current SubState: Accelerate
Error: Can only press Left Pedal thrice when in Decelerate State.
Unable to move.
Current State: Move Backward
Current SubState: Accelerate
Transitioning from Accelerate State to Decelerate State...
Current State: Move Backward
Current SubState: Decelerate
Transitioning from Decelerate State to At Rest State...
Current State: At Rest
Current SubState: None
BUILD SUCCESSFUL (total time: 0 seconds)
```

The status bar at the bottom right indicates the date as 23-11-2023 and the time as 19:04.

Test case 1:

The screenshot shows the Apache NetBeans IDE interface with the 'RoverTest.java' file open in the code editor. The Projects tab shows the 'RoverTest' project structure. The Test Results tab displays the outcome of the 'testPressRightPedal' test:

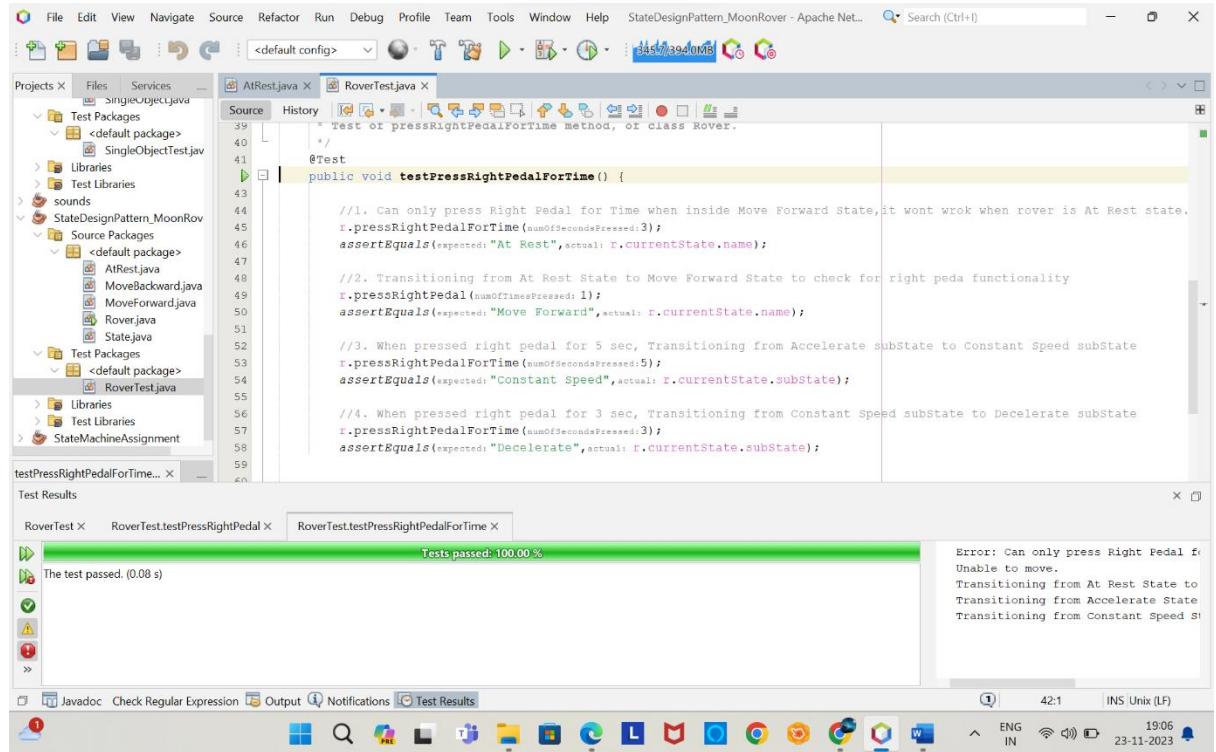
Tests passed: 100.00 %

The test passed. (0.089 s)

Error: Can only press Right Pedal or
Unable to move.
Transitioning from At Rest State to

The status bar at the bottom right indicates the date as 23-11-2023 and the time as 23:11.

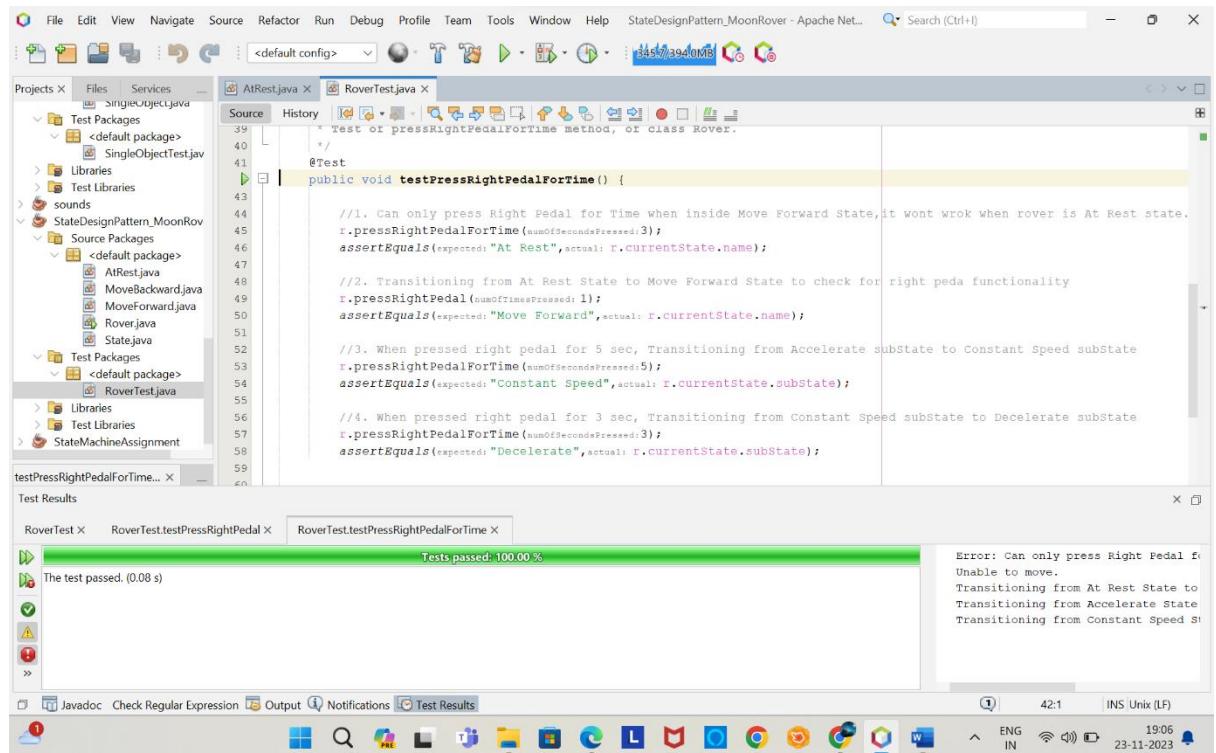
Test case 2:



The screenshot shows the Eclipse IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and run.
- Project Explorer:** Shows the project structure with packages like SingleObjectJava, Test Packages, Libraries, and StateDesignPattern_MoonRov.
- Source Editor:** Displays the code for `RoverTest.java` containing a single test method `testPressRightPedalForTime()`. The code includes four test cases for pressing the right pedal for different durations to check state transitions.
- Test Results:** Shows the test results for `RoverTest.testPressRightPedalForTime` with a status of "Tests passed: 100.00 %".
- Bottom Status Bar:** Shows system information including battery level (345/3940MB), network (WIFI), and date/time (23-11-2023).

Test case 3:



The screenshot shows the Eclipse IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help.
- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and run.
- Project Explorer:** Shows the project structure with packages like SingleObjectJava, Test Packages, Libraries, and StateDesignPattern_MoonRov.
- Source Editor:** Displays the code for `RoverTest.java` containing a single test method `testPressRightPedalForTime()`. The code includes four test cases for pressing the right pedal for different durations to check state transitions.
- Test Results:** Shows the test results for `RoverTest.testPressRightPedalForTime` with a status of "Tests passed: 100.00 %".
- Console:** Shows an error message: "Error: Can only press Right Pedal f... Unable to move. Transitioning from At Rest State to Transitioning from Accelerate State Transitioning from Constant Speed S..."
- Bottom Status Bar:** Shows system information including battery level (345/3940MB), network (WIFI), and date/time (23-11-2023).

Test Case 4:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and StateDesignPattern_MoonRover - Apache Net... A search bar is at the top right. Below the menu is a toolbar with various icons. The left sidebar shows the Projects tree, which includes a SingleObjectJava project with a Test Packages folder containing SingleObjectTest.java, and a StateDesignPattern_MoonRov project with Source Packages (containing AIRest.java, MoveBackward.java, MoveForward.java, Rover.java, State.java) and Test Packages (containing RoverTest.java). The main editor window displays the code for RoverTest.java, specifically the testPressLeftPedalForTime() method. The code performs three tests: transitioning from At Rest state to Accelerate state, then from Accelerate state to Constant Speed state, and finally from Constant Speed state to Move Backward state. The code uses assertEquals() assertions to check the current state name and substate. The status bar at the bottom shows the build output: 2425/3940MB.

```
    * Test of pressLeftPedalForTime method, of class Rover.  
    */  
    @Test  
    public void testPressLeftPedalForTime() {  
        //1. Rover should always start in "At Rest" state  
        assertEquals(expected: "At Rest", actual: r.currentState.name);  
  
        //2. When at rest, pressing left pedal for 5 sec  
        r.pressLeftPedalForTime(numOfSecondsPressed: 5);  
        assertEquals(expected: "Move Backward", actual: r.currentState.name);  
  
        //3. When at Move Backward state, pressing left pedal for 3 sec to move rover in constant backward speed subState  
        r.pressLeftPedalForTime(numOfSecondsPressed: 3);  
        assertEquals(expected: "Constant Speed", actual: r.currentState.subState);  
  
        //3. When at Move Backward state, pressing left pedal for 3 sec to accelerate the rover in forward direction  
        r.pressLeftPedalForTime(numOfSecondsPressed: 3);  
        assertEquals(expected: "Accelerate", actual: r.currentState.subState);  
    }  
  
The test passed. (0.078 s)  
Tests passed: 100.00 %
```

Transitioning from At Rest State to Accelerate State
Transitioning from Accelerate State to Constant Speed State
Transitioning from Constant Speed State to Move Backward State

Test case 5:

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and StateDesignPattern_MoonRover - Apache Net... A search bar is at the top right. Below the menu is a toolbar with various icons. The left sidebar shows the Projects tree, which includes a SingleObjectJava project with a Test Packages folder containing SingleObjectTest.java, and a StateDesignPattern_MoonRov project with Source Packages (containing AIRest.java, MoveBackward.java, MoveForward.java, Rover.java, State.java) and Test Packages (containing RoverTest.java). The main editor window displays the code for RoverTest.java, specifically the testPrintStateAndSubState() and testMain() methods. The code performs two tests: printing the current state and substate, and running the main method. The code uses assertEquals() assertions to check the current state name and substate. The status bar at the bottom shows the build output: 2425/3940MB.

```
    * When at Move Backward state, pressing left pedal for 3 sec to accelerate the rover in forward direction  
    r.pressLeftPedalForTime(numOfSecondsPressed: 3);  
    assertEquals(expected: "Accelerate", actual: r.currentState.subState);  
}  
  
    /**  
     * Test of printStateAndSubState method, of class Rover.  
     */  
    @Test  
    public void testPrintStateAndSubState() {  
    }  
  
    /**  
     * Test of main method, of class Rover.  
     */  
    @Test  
    public void testMain() {  
    }  
  
The test passed. (0.084 s)  
Tests passed: 100.00 %
```

114:1 INS Unix (LF)

Test case 6:

```

104     /**
105      * When Move Backward state, pressing left pedal for 3 sec to accelerate the rover in forward direction
106      */
107     @Test
108     public void testPressLeftPedal() {
109         /*
110          * Test of printStateAndSubState method, of class Rover.
111          */
112         @Test
113         public void testPrintStateAndSubState() {
114             /*
115              * Test of main method, of class Rover.
116              */
117             @Test
118             public void testMain() {
119                 /*
120                   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
121                   * Click nbfs://nbhost/SystemFileSystem/Templates/UnitTests/JUnit4TestClass.java to edit this template
122               */
123           }
124       }

```

Test Results

- RoverTest.testPressRightPedal X
- RoverTest.testPressRightPedalForTime X
- RoverTest.testPressLeftPedal X
- RoverTest.testPressLeftPedalForTime X
- RoverTest.testPrintStateAndSubState X
- RoverTest.testMain X

Tests passed: 100.00 %

The test passed. (0.07 s)

JavaDoc Check Regular Expression Output Notifications Test Results

121:1 19:08 23-11-2023 ENG IN

All Test cases:

```

1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/UnitTests/JUnit4TestClass.java to edit this template
4   */
5
6 import org.junit.Test;
7 import static org.junit.Assert.*;
8
9 /**
10  * @author Namitha
11  * @version 1.0
12  */
13
14 public class RoverTest {
15
16     public RoverTest() {
17     }
18
19     /**
20      * Test of pressRightPedal method, of class Rover.
21      */
22     Rover r = new Rover();

```

Test Results

- ShapeMakerTest.testDrawSquare X
- RoverTest X

Tests passed: 100.00 %

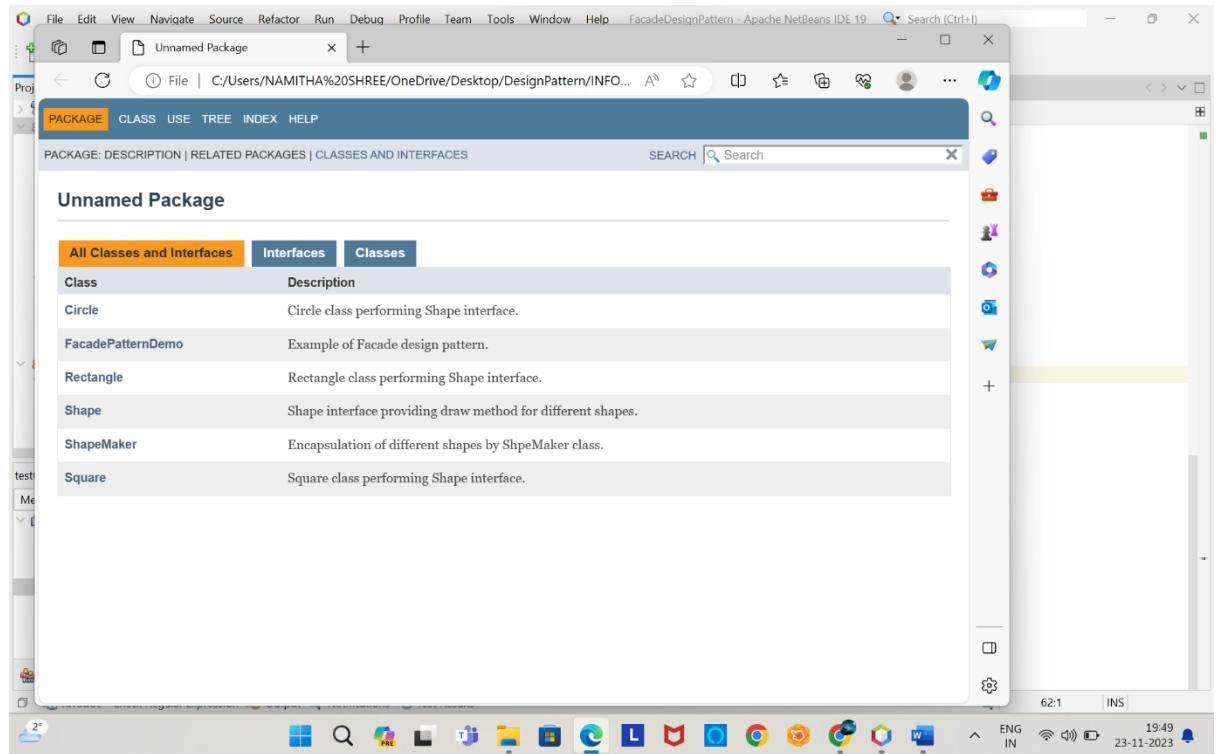
All 6 tests passed. (0.152 s)

Transitioning from At Rest State to Accelerate State
Transitioning from Constant Speed State
Error: Can only press Left Pedal when Unable to move.
Transitioning from At Rest State to Decelerate State
Transitioning from Decelerate State to Accelerate State
Error: Can only press Right Pedal or

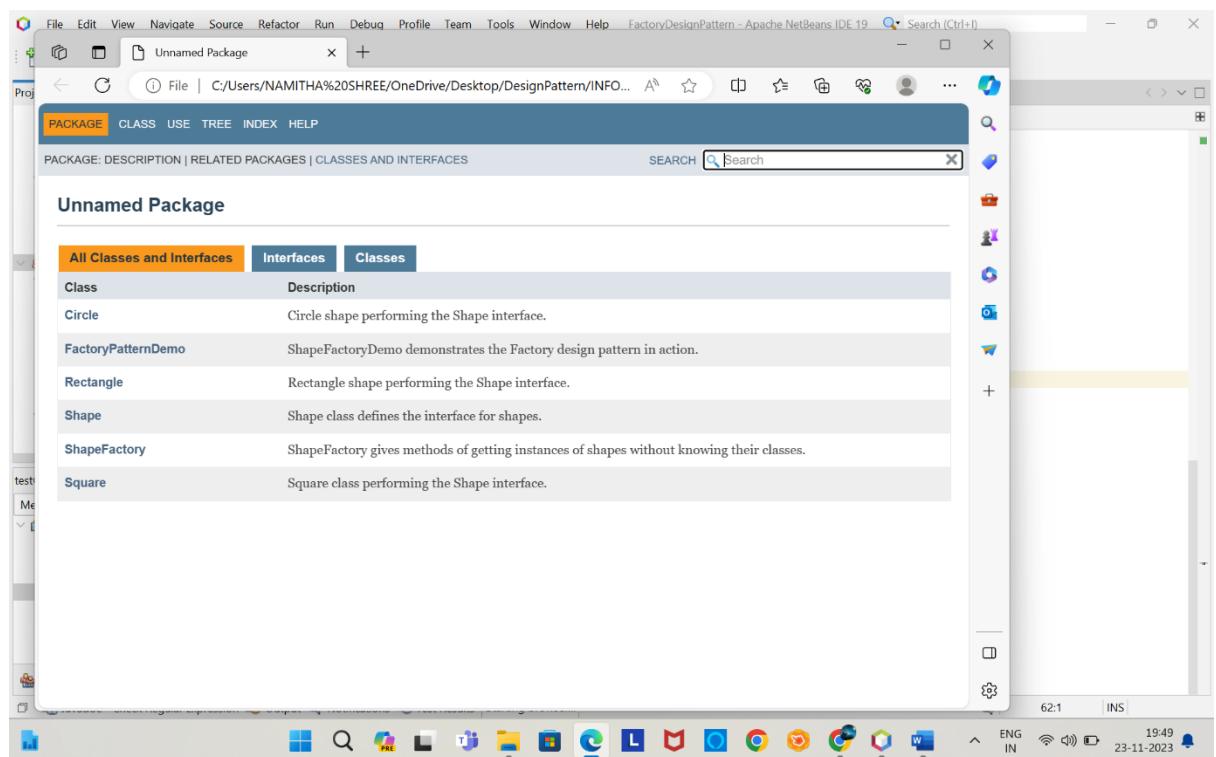
JavaDoc Check Regular Expression Output Notifications Test Results

1:1 19:05 23-11-2023 ENG IN

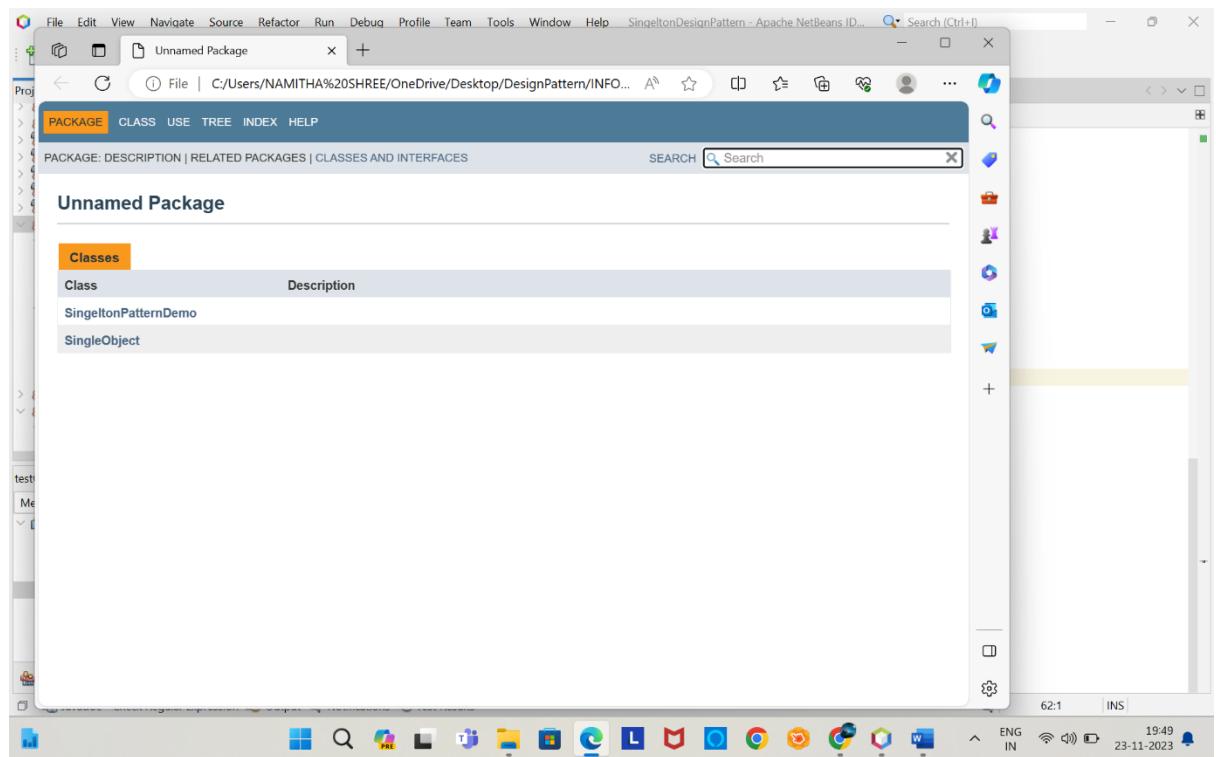
FAÇADE DESIGN PATTERN-JAVADOC



FACTORY DESIGN PATTERN-JAVADOC



SINGLETON PATTERN- JAVADOC



STATE DESIGN PATTERN-MOON ROVER- JAVADOC

