```
/**
 * @author Harley Phung
 * Project 4 Tester - create a game called Tsuro for 2 players.
 */
import org.junit.*;
import static org.junit.Assert.*;
import java.lang.ExceptionInInitializerError;
import javafx.scene.paint.Color;

public class TsuroTester{
    Tsuro test1 = new Tsuro();

    /**
     * Test the getNumRow() method
     */
    @Test
    public void testGetNumRow() {
        assertEquals(6, test1.getNumRow());
    }

    /**
     * Test the getNumCol() method
     */
    @Test
    public void testGetNumCol() {
        assertEquals(6, test1.getNumCol());
    }

    /**
     * Test the getHandCard() method
     */
    @Test
    public void testGetHandCard() {
        assertEquals(3, test1.getHandCard());
    }

    /**
     * Test the getNumPlayers() method
     */
    @Test
    public void testGetNumPlayers() {
        assertEquals(2, test1.getNumPlayers());
    }

    /**
     * Test the getChosenButton1() method
     */
    @Test
    public void testGetChosenButton1() {
        assertEquals(null, test1.getChosenButton1());
    }

    /**
     * Test the setChosenButton1() method
     */
    @Test
    public void testSetChosenButton1() {
        try {
            TsuroButton chosenButton1_1 = new TsuroButton(50, 50);
```

```
        test1.setChosenButton1(chosenButton1_1);
        assertEquals(chosenButton1_1, test1.getChosenButton1());
    }
    catch (ExceptionInInitializerError e) {
        ;
    }
    catch(NoClassDefFoundError e) {
        ;
    }
}

/**
 * Test the getChosenButton2() method
 */
@Test
public void testGetChosenButton2() {
    assertEquals(null, test1.getChosenButton2());
}

/**
 * Test the setChosenButton1() method
 */
@Test
public void testSetChosenButton2() {
    try {
        TsuroButton chosenButton2_1 = new TsuroButton(50, 50);
        test1.setChosenButton2(chosenButton2_1);
        assertEquals(chosenButton2_1, test1.getChosenButton2());
    }
    catch (ExceptionInInitializerError e) {
        ;
    }
    catch(NoClassDefFoundError e) {
        ;
    }
}


/**
 * Test the getSelectedBoard1() method
 */
@Test
public void testGetSelectedBoard1() {
    assertEquals(null, test1.getSelectedBoard1());
}

/**
 * Test the setSelectedBoard1() method
 */
@Test
public void testSetSelectedBoard1() {
    try {
        TsuroButton selectedBoard1_1 = new TsuroButton(50, 50);
        test1.setSelectedBoard1(selectedBoard1_1);
        assertEquals(selectedBoard1_1, test1.getSelectedBoard1());
    }
    catch (ExceptionInInitializerError e) {
        ;
    }
```

```
    catch(NoClassDefFoundError e) {
        ;
    }
}

/**
 * Test the getSelectedBoard2() method
 */
@Test
public void testGetSelectedBoard2() {
    assertEquals(null, test1.getSelectedBoard2());
}

/**
 * Test the setSelectedBoard1() method
 */
@Test
public void testSetSelectedBoard2() {
    try {
        TsuroButton selectedBoard2_1 = new TsuroButton(50, 50);
        test1.setSelectedBoard2(selectedBoard2_1);
        assertEquals(selectedBoard2_1, test1.getSelectedBoard2());
    }
    catch (ExceptionInInitializerError e) {
        ;
    }
    catch(NoClassDefFoundError e) {
        ;
    }
}

/**
 * Test the getCurrentBlue() method
 */
@Test
public void testGetCurrentBlue() {
    assertEquals(-1, test1.getCurrentBlue());
}

/**
 * Test the setCurrentBlue() method
 */
@Test
public void testSetCurrentBlue() {
    test1.setCurrentBlue(6);
    assertEquals(6, test1.getCurrentBlue());
    test1.setCurrentBlue(2);
    assertEquals(2, test1.getCurrentBlue());
}

/**
 * Test the getCurrentGreen() method
 */
@Test
public void testGetCurrentGreen() {
    assertEquals(-1, test1.getCurrentGreen());
}

/**
```

```
 * Test the setCurrentBlue() method
 */
@Test
public void testSetCurrentGreen() {
    test1.setCurrentGreen(3);
    assertEquals(3, test1.getCurrentGreen());
    test1.setCurrentGreen(0);
    assertEquals(0, test1.getCurrentGreen());
}

/**
 * Test the thisPosition() method
 */
@Test
public void testThisPosition() {
    try {
        TsuroButton boardButton1 = new TsuroButton(50,50);
        TsuroButton boardButton3 = new TsuroButton(50,50);
        TsuroButton[][] boardButton2 = new TsuroButton[6][6];
        TsuroButton[][] boardButton4 = new TsuroButton[7][2];
        boardButton2[3][4] = boardButton1;
        boardButton4[0][1] = boardButton3;
        assertEquals(3, test1.thisPosition(boardButton1)[0]);
        assertEquals(4, test1.thisPosition(boardButton1)[1]);
        assertEquals(0, test1.thisPosition(boardButton3)[0]);
        assertEquals(1, test1.thisPosition(boardButton3)[1]);
    }
    catch (NoClassDefFoundError e) {
        ;
    }
}

/**
 * Test the rotateChosenButton() method
 */
@Test
public void testRotateChosenButton() {
    try {
        TsuroButton button = new TsuroButton(50, 50);
        TsuroButton button2 = new TsuroButton(50, 50);
        int[] path1 = {2, 4, 6, 1, 7, 0, 3, 5};
        int[] path2 = {4, 6, 0, 3, 2, 1, 7, 5};
        int[] outputPath1 = {5, 6, 1, 3, 0, 2, 4, 7};
        int[] outputPath2 = {6, 1, 2, 4, 5, 3, 1, 7};
        button.setConnections(path1);
        button.setConnections(path2);
        test1.rotateChosenButton(0, button);
        test1.rotateChosenButton(1, button2);
        assertEquals(outputPath1, button.getConnections());
        assertEquals(outputPath2, button2.getConnections());
    }
    catch (NoClassDefFoundError e) {
        ;
    }
}
/**
 * Test the removeAllStone() method
 */
@Test
```

```
    public void testRemoveAllStone() {
        try {
            TsuroButton button1 = new TsuroButton(50,50);
            button1.addStone(Color.BLUE, 6);
            button1.addStone(Color.BLUE, 3);
            button1.addStone(Color.BLUE, 0);
            test1.removeAllStones(button1);
            assertEquals(null, button1);
            assertEquals(null, button1);
            assertEquals(null, button1);
        }
        catch(ExceptionInInitializerError e) {
            ;
        }
        catch(NoClassDefFoundError er) {
            ;
        }
    }

    /**
     * Test the stoneCollide() method
     */
    @Test
    public void testStoneCollide() {
        try {
            TsuroButton button1 = new TsuroButton(50, 50);
            TsuroButton button2 = new TsuroButton(50, 50);
            test1.thisPosition(button1)[0] = 3;
            test1.thisPosition(button1)[1] = 3;
            test1.thisPosition(button2)[0] = 4;
            test1.thisPosition(button2)[1] = 3;
            //When button 1 is above button2, there's only 2 possibilities to
collide.
            test1.setCurrentBlue(4);
            test1.setCurrentGreen(0);
            assertTrue(test1.stoneCollide());

            test1.setCurrentBlue(5);
            test1.setCurrentGreen(1);
            assertTrue(test1.stoneCollide());

            //Other combination of stones' positions are false
            //1. If they match their positions 6-2, 7-3, 0-4, 1-5, 2-6, 3-7
            test1.setCurrentBlue(6);
            test1.setCurrentGreen(2);
            assertFalse(test1.stoneCollide());

            test1.setCurrentBlue(7);
            test1.setCurrentGreen(3);
            assertFalse(test1.stoneCollide());

            test1.setCurrentBlue(0);
            test1.setCurrentGreen(4);
            assertFalse(test1.stoneCollide());

            test1.setCurrentBlue(1);
            test1.setCurrentGreen(5);
            assertFalse(test1.stoneCollide());
```

```
        test1.setCurrentBlue(2);
        test1.setCurrentGreen(6);
        assertFalse(test1.stoneCollide());

        test1.setCurrentBlue(3);
        test1.setCurrentGreen(7);
        assertFalse(test1.stoneCollide());

        //If the 2 positions are not match
        test1.setCurrentBlue(6);
        test1.setCurrentGreen(5);
        assertFalse(test1.stoneCollide());

        test1.setCurrentBlue(2);
        test1.setCurrentGreen(7);
        assertFalse(test1.stoneCollide());

        test1.setCurrentBlue(1);
        test1.setCurrentGreen(0);
        assertFalse(test1.stoneCollide());
    }
    catch (NoClassDefFoundError e) {
        ;
    }
}

/**
 * Test the outOfBoard() method
 */
@Test
public void testOutOfBoard() {
    try {
        TsuroButton button1 = new TsuroButton(50, 50);
        test1.thisPosition(button1)[0] = 0;
        test1.thisPosition(button1)[1] = 2;

        test1.setCurrentBlue(0);
        assertTrue(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));

        test1.setCurrentBlue(1);
        assertTrue(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));

        test1.setCurrentBlue(2);
        assertTrue(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));

        test1.setCurrentBlue(3);
        assertTrue(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));

        test1.setCurrentBlue(4);
        assertFalse(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));

        test1.setCurrentBlue(5);
        assertFalse(test1.outOfBoard(0));
        assertFalse(test1.outOfBoard(1));
```

```
            test1.setCurrentBlue(6);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));

            test1.setCurrentBlue(7);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));

            test1.setCurrentGreen(0);
            assertFalse(test1.outOfBoard(0));
            assertTrue(test1.outOfBoard(1));

            test1.setCurrentGreen(1);
            assertFalse(test1.outOfBoard(0));
            assertTrue(test1.outOfBoard(1));

            test1.setCurrentGreen(2);
            assertFalse(test1.outOfBoard(0));
            assertTrue(test1.outOfBoard(1));

            test1.setCurrentGreen(3);
            assertFalse(test1.outOfBoard(0));
            assertTrue(test1.outOfBoard(1));

            test1.setCurrentGreen(4);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));

            test1.setCurrentGreen(5);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));

            test1.setCurrentGreen(6);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));

            test1.setCurrentGreen(7);
            assertFalse(test1.outOfBoard(0));
            assertFalse(test1.outOfBoard(1));
        }
        catch (NoClassDefFoundError e) {
            ;
        }
    }

    /**
     * Test the findNextTile() method
     */
    @Test
    public void testFindNextTile() {
        //Test when the current stone position is 0
        assertEquals(2, test1.findNextTile(3, 3, 0)[0]);
        assertEquals(3, test1.findNextTile(3, 3, 0)[1]);

        //Test when the current stone position is 1
        assertEquals(2, test1.findNextTile(3, 3, 1)[0]);
        assertEquals(3, test1.findNextTile(3, 3, 1)[1]);
```

```
        //Test when the current stone position is 2
        assertEquals(3, test1.findNextTile(3, 3, 2)[0]);
        assertEquals(4, test1.findNextTile(3, 3, 2)[1]);

        //Test when the current stone position is 3
        assertEquals(3, test1.findNextTile(3, 3, 3)[0]);
        assertEquals(4, test1.findNextTile(3, 3, 3)[1]);

        //Test when the current stone position is 4
        assertEquals(4, test1.findNextTile(3, 3, 4)[0]);
        assertEquals(3, test1.findNextTile(3, 3, 4)[1]);

        //Test when the current stone position is 5
        assertEquals(4, test1.findNextTile(3, 3, 5)[0]);
        assertEquals(3, test1.findNextTile(3, 3, 5)[1]);

        //Test when the current stone position is 6
        assertEquals(3, test1.findNextTile(3, 3, 6)[0]);
        assertEquals(2, test1.findNextTile(3, 3, 6)[1]);

        //Test when the current stone position is 7
        assertEquals(3, test1.findNextTile(3, 3, 7)[0]);
        assertEquals(2, test1.findNextTile(3, 3, 7)[1]);
    }
}
```