```java
package tests;

import static org.junit.Assert.*;

import java.util.Set;

import org.junit.BeforeClass;
import org.junit.Test;

import clueGame.Board;
import experiment.BoardCell;

public class testAdjTargets
{

  private static Board board;
  @BeforeClass
  public static void setUp() {
    // Board is singleton, get the only instance
    board = Board.getInstance();
    // set the file names to use my config files
    board.setConfigFiles("Clue_Board_Layout_Langfield_Prather.csv", "LegendForClueLayout.txt");
    // Initialize will load BOTH config files
    board.initialize();
  }

  //Check that adjacencies for walkway are only on walkway unless
  // door that is enterable
  @Test
  public void testWalkwayAdj()
  {
    Set<clueGame.BoardCell> cells = board.getAdjList(13, 15);
    for (clueGame.BoardCell tempCell : cells)
    {
      assertFalse(tempCell.isDoorway());
    }
    assertTrue(cells.contains(board.getCellAt(14, 15)));
  }

  // Test that in room there should be no adjacencies
  @Test
  public void testAdjInRoom()
  {
    Set<clueGame.BoardCell> cells = board.getAdjList(14, 19);
    assertTrue(cells.isEmpty());

  }
  // Check cant go over board
  @Test
  public void testEdgeOfBoard()
  {
    Set<clueGame.BoardCell> cells = board.getAdjList(20, 22);
    for(clueGame.BoardCell cell: cells) {
      if(cell == null) {
        assertFalse(cells.contains(board.getCellAt(21, 22)));
        assertFalse(cells.contains(board.getCellAt(20, 23)));

        cells = board.getAdjList(0, 9);
        assertFalse(cells.contains(board.getCellAt(40, 40)));
```

```
60          assertFalse(cells.contains(board.getCellAt(-1, 9)));
61        }
62      }
63
64    }
65
66    //Check that we cant go into a room without a doorway
67    @Test
68    public void testNotDoorway()
69    {
70      Set<clueGame.BoardCell> cells = board.getAdjList(3, 5);
71      assertFalse(cells.contains(board.getCellAt(3, 4)));
72
73      cells = board.getAdjList(3, 13);
74      assertFalse(cells.contains(board.getCellAt(3, 14)));
75
76    }
77
78    // Check that we can go into doorway
79    @Test
80    public void testEnterDoorAdj()
81    {
82      // Up
83      Set<clueGame.BoardCell> cells = board.getAdjList(11, 18);
84      assertTrue(cells.contains(board.getCellAt(12,18)));
85
86      //Down
87      cells = board.getAdjList(5, 11);
88      assertTrue(cells.contains(board.getCellAt(4,11)));
89
90      //Right
91      cells = board.getAdjList(18, 4);
92      assertTrue(cells.contains(board.getCellAt(18,3)));
93
94      //Left
95      cells = board.getAdjList(20, 6);
96      assertTrue(cells.contains(board.getCellAt(20, 7)));
97
98    }
99
100   // Check locations that are doorways should only let you in to room
101   @Test
102   public void testDoorAdj()
103   {
104     Set<clueGame.BoardCell> cells = board.getAdjList(4, 11);
105     assertTrue(cells.contains(board.getCellAt(5,11)));
106     assertEquals(cells.size(), 1);
107
108     cells = board.getAdjList(6, 19);
109     assertTrue(cells.contains(board.getCellAt(7,19)));
110     assertEquals(cells.size(), 1);
111
112   }
113
114   // Check targets along walkways
115   @Test
116   public void testWalkwayTargets()
117   {
118     board.calcTargets(15, 9, 2);
119     Set<clueGame.BoardCell> cells = board.getTargets();
```

```
120     assertTrue(cells.contains(board.getCellAt(16, 10)));
121     assertFalse(cells.contains(board.getCellAt(17,9)));
122 
123     board.calcTargets(5, 13, 2);
124     cells = board.getTargets();
125     assertTrue(cells.contains(board.getCellAt(7, 13)));
126     assertFalse(cells.contains(board.getCellAt(5,15)));
127 
128     board.calcTargets(9, 19, 4);
129     cells = board.getTargets();
130     assertTrue(cells.contains(board.getCellAt(9, 15)));
131     assertFalse(cells.contains(board.getCellAt(13,19)));
132 
133     board.calcTargets(3, 5, 2);
134     cells = board.getTargets();
135     assertTrue(cells.contains(board.getCellAt(5, 5)));
136     assertFalse(cells.contains(board.getCellAt(3,3)));
137 
138   }
139 
140   // Test that the user can enter a room with target
141   @Test
142   public void testEnterRoomTarget()
143   {
144     board.calcTargets(8, 4, 3);
145     Set<clueGame.BoardCell> cells = board.getTargets();
146     assertTrue(cells.contains(board.getCellAt(8, 3)));
147     assertEquals(cells.size(), 10);
148 
149     board.calcTargets(9, 16, 1);
150     cells = board.getTargets();
151     assertTrue(cells.contains(board.getCellAt(8, 16)));
152     assertEquals(cells.size(), 4);
153 
154   }
155 
156   // Check targets when leaving room
157   @Test
158   public void testLeaveRoomTargets()
159   {
160     board.calcTargets(14, 5, 1);
161     Set<clueGame.BoardCell> cells = board.getTargets();
162     assertTrue(cells.contains(board.getCellAt(14, 6)));
163 
164     board.calcTargets(4, 11, 2);
165     cells = board.getTargets();
166     assertTrue(cells.contains(board.getCellAt(6, 11)));
167 
168   }
169 
170 
171 }
172 
```