CS472 Lab – Dynamic Analysis

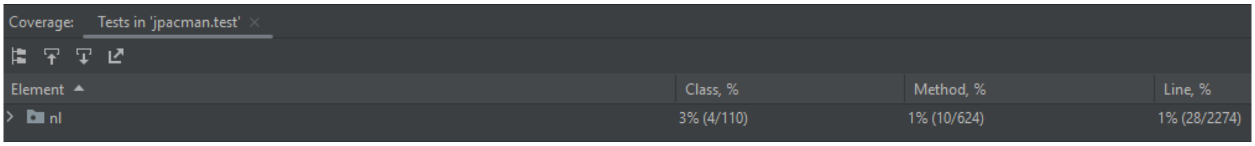
Task 2.1:

The methods chosen to test was whether the player collides with a pellet, collides with ghost, and a direction test.

Task 3:

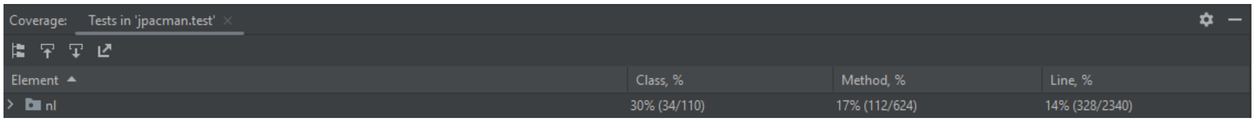
The coverages were not like from JaCoCo were not similar to the ones that were from IntelliJ. I believe this is because the tests done with IntelliJ did not cover as much as the ones from the original report. The source code visualization from JaCoCo helped a lot and with the visualizations I much prefer JaCoCo's report because of the visuals that it provides.

Before added tests:



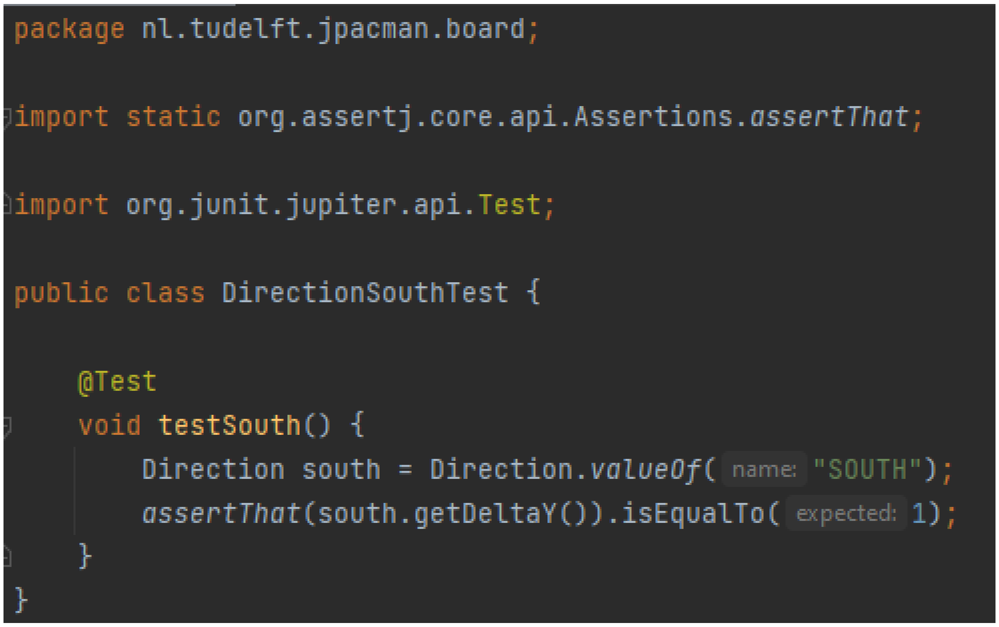
Element	Class, %	Method, %	Line, %
nl	3% (4/110)	1% (10/624)	1% (28/2274)

After added tests:



Element	Class, %	Method, %	Line, %
nl	30% (34/110)	17% (112/624)	14% (328/2340)

Code Snippets:



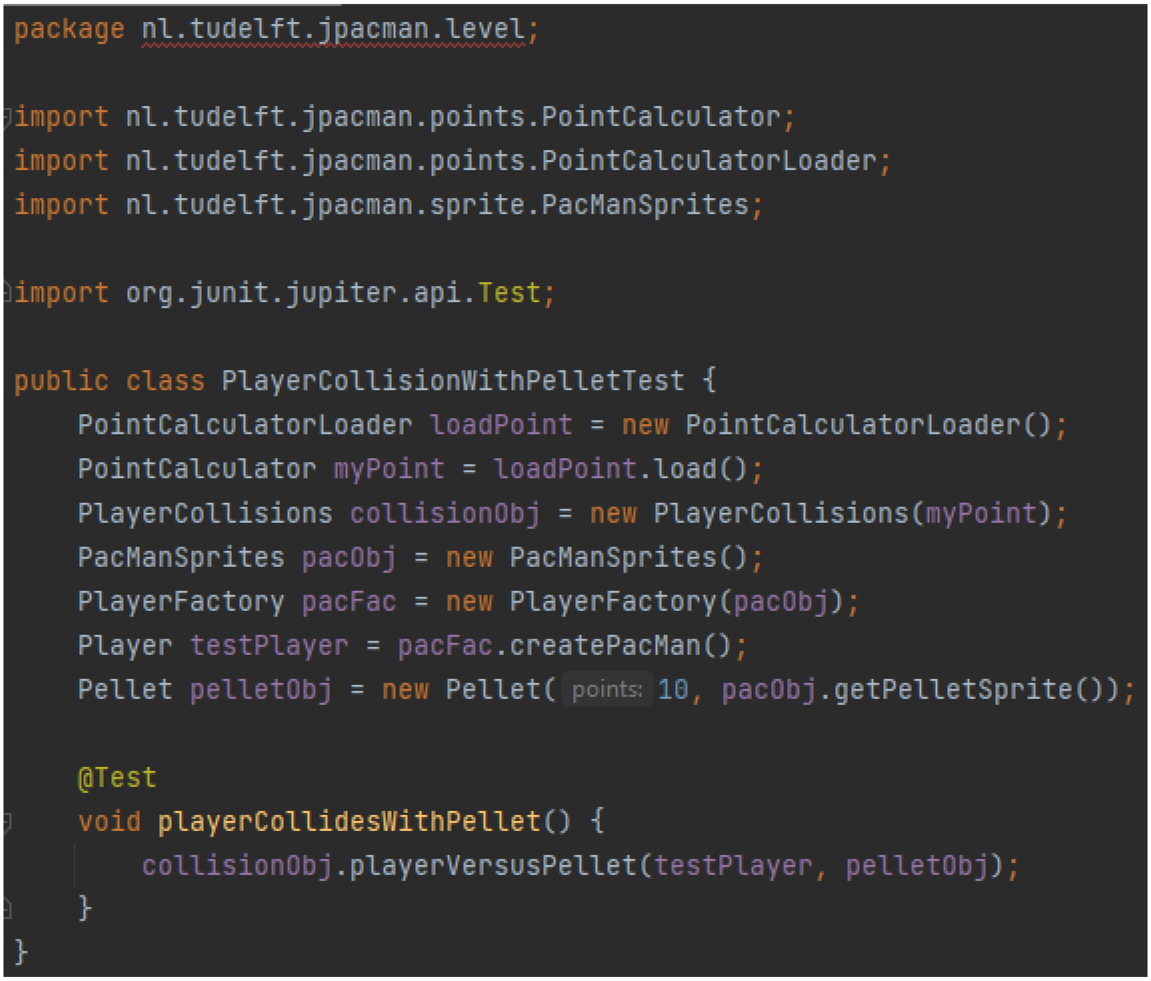
```
package nl.tudelft.jpacman.board;

import static org.assertj.core.api.Assertions.assertThat;

import org.junit.jupiter.api.Test;

public class DirectionSouthTest {

    @Test
    void testSouth() {
        Direction south = Direction.valueOf( name: "SOUTH");
        assertThat(south.getDeltaY()).isEqualTo( expected: 1);
    }
}
```



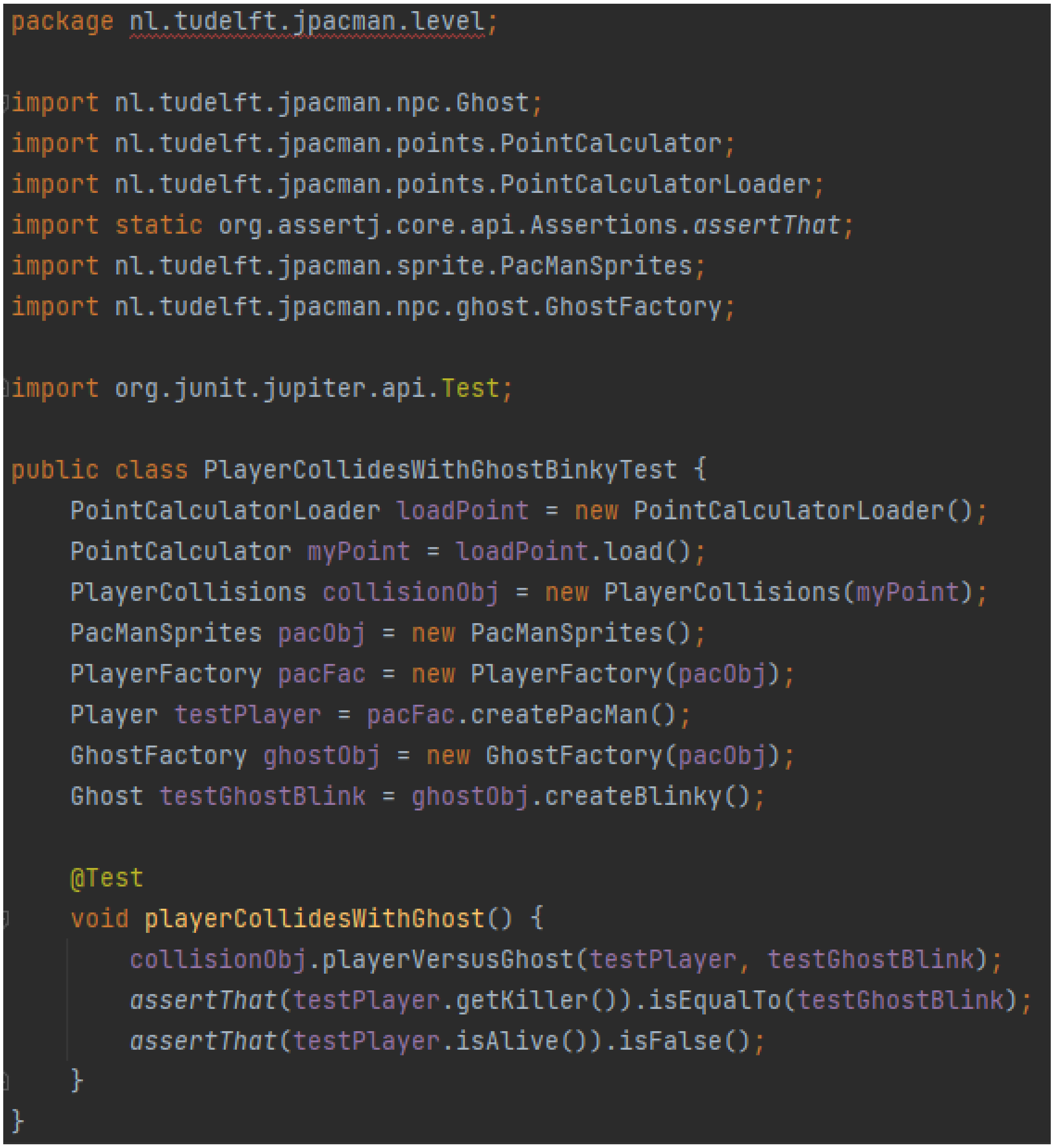
```
package nl.tudelft.jpacman.level;

import nl.tudelft.jpacman.points.PointCalculator;
import nl.tudelft.jpacman.points.PointCalculatorLoader;
import nl.tudelft.jpacman.sprite.PacManSprites;

import org.junit.jupiter.api.Test;

public class PlayerCollisionWithPelletTest {
    PointCalculatorLoader loadPoint = new PointCalculatorLoader();
    PointCalculator myPoint = loadPoint.load();
    PlayerCollisions collisionObj = new PlayerCollisions(myPoint);
    PacManSprites pacObj = new PacManSprites();
    PlayerFactory pacFac = new PlayerFactory(pacObj);
    Player testPlayer = pacFac.createPacMan();
    Pellet pelletObj = new Pellet( points: 10, pacObj.getPelletSprite());

    @Test
    void playerCollidesWithPellet() {
        collisionObj.playerVersusPellet(testPlayer, pelletObj);
    }
}
```



```java
package nl.tudelft.jpacman.level;

import nl.tudelft.jpacman.npc.Ghost;
import nl.tudelft.jpacman.points.PointCalculator;
import nl.tudelft.jpacman.points.PointCalculatorLoader;
import static org.assertj.core.api.Assertions.assertThat;
import nl.tudelft.jpacman.sprite.PacManSprites;
import nl.tudelft.jpacman.npc.ghost.GhostFactory;

import org.junit.jupiter.api.Test;

public class PlayerCollidesWithGhostBinkyTest {
    PointCalculatorLoader loadPoint = new PointCalculatorLoader();
    PointCalculator myPoint = loadPoint.load();
    PlayerCollisions collisionObj = new PlayerCollisions(myPoint);
    PacManSprites pacObj = new PacManSprites();
    PlayerFactory pacFac = new PlayerFactory(pacObj);
    Player testPlayer = pacFac.createPacMan();
    GhostFactory ghostObj = new GhostFactory(pacObj);
    Ghost testGhostBlink = ghostObj.createBlinky();

    @Test
    void playerCollidesWithGhost() {
        collisionObj.playerVersusGhost(testPlayer, testGhostBlink);
        assertThat(testPlayer.getKiller()).isEqualTo(testGhostBlink);
        assertThat(testPlayer.isAlive()).isFalse();
    }
}
```