

**Report by:** Avery Nguyen, 5006145789, nguyea26@unlv.nevada.edu

**Forked Repository Link:** <https://github.com/akummi/jpacman/branches>

**Main Repository:** <https://github.com/CivBuilder/cs472project>

**Task 1:** Screenshot of tests run with no tests added

nl	3% (4/110)	1% (10/624)	1% (28/2274)
tudelft	3% (4/110)	1% (10/624)	1% (28/2274)
jpacman	3% (4/110)	1% (10/624)	1% (28/2274)
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	0% (0/26)	0% (0/156)	0% (0/690)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	0% (0/4)	0% (0/14)	0% (0/38)
sprite	0% (0/12)	0% (0/90)	0% (0/238)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurator	0% (0/1)	0% (0/2)	0% (0/4)

**Task 2:** Added isAlive() test

nl	16% (18/...	9% (60/624)	8% (190/23...
tudelft	16% (18/...	9% (60/624)	8% (190/23...
jpacman	16% (18/...	9% (60/624)	8% (190/23...
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	15% (4/26)	6% (10/156)	3% (26/700)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	0% (0/4)	0% (0/14)	0% (0/38)
sprite	83% (10/...	44% (40/90)	52% (136/2...
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurator	0% (0/1)	0% (0/2)	0% (0/4)

```

package nl.tudelft.jpacman.level;

import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;

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

no usages new *
public class PlayerTest {

    1 usage
    private static final PacManSprites SPRITE_STORE = new PacManSprites();
    1 usage
    private PlayerFactory Factory = new PlayerFactory(SPRITE_STORE);
    5 usages
    private Player ThePlayer = Factory.createPacMan();

    no usages new *
    @Test
    void testAlive(){
        assertThat(ThePlayer.isAlive()).isEqualTo( expected: true);
    }
}

```

## Task 2.1: Added my own tests

### Test 1:

Code for method test playerVersusPellet

Method in: src/main/java/nl/tudelft/jpacman/level/PlayerCollisions.java

```

1 package nl.tudelft.jpacman.level;
2
3 import nl.tudelft.jpacman.points.DefaultPointCalculator;
4 import nl.tudelft.jpacman.sprite.EmptySprite;
5 import nl.tudelft.jpacman.sprite.PacManSprites;
6 import org.junit.jupiter.api.Test;
7 import nl.tudelft.jpacman.points.PointCalculator;
8
9 import static org.assertj.core.api.Assertions.assertThat;
10
11 no usages new *
12 public class PlayerCollisionsTest{
13     1 usage
14     private static final EmptySprite emptySprite = new EmptySprite();
15     1 usage
16     Pellet pellet = new Pellet( points: 1,emptySprite);
17     1 usage
18     private static final PacManSprites SPRITE_STORE = new PacManSprites();
19     1 usage
20     private PlayerFactory Factory = new PlayerFactory(SPRITE_STORE);
21     3 usages
22     private Player ThePlayer = Factory.createPacMan();
23     1 usage
24     private PointCalculator calculator = new DefaultPointCalculator();
25     1 usage
26     PlayerCollisions collisions = new PlayerCollisions(calculator);
27
28     no usages new *
29     @Test
30     void testCollision(){
31         int oldscore = ThePlayer.getScore(); //saves old score
32         collisions.playerVersusPellet(ThePlayer, pellet); //player hits pellet
33         assertThat( actual: ThePlayer.getScore() > oldscore); //if player hits pellet score should increase
34     }
35 }
36

```

## Coverage after method test playerVersusPellet

nl	21% (24/110)	11% (74/624)	9% (228/2284)
tudelft	21% (24/110)	11% (74/624)	9% (228/2284)
jpacman	21% (24/110)	11% (74/624)	9% (228/2284)
board	20% (4/20)	11% (12/106)	12% (34/272)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	12% (20/156)	7% (52/684)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	50% (2/4)	14% (2/14)	14% (6/42)
sprite	83% (10/12)	44% (40/90)	52% (136/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurator	0% (0/1)	0% (0/2)	0% (0/4)

## Test 2:

Code for method testing pacmanMoved, consumedAPellet, getScore.

Methods in: src/main/java/nl/tudelft/jpacman/points/DefaultPointCalculator.java

```
no usages new *
15 public class DefaultPointCalculatorTest {
16     1 usage
17     private static final EmptySprite emptySprite = new EmptySprite();
18     1 usage
19     Pellet pellet = new Pellet( points: 1,emptySprite);
20     1 usage
21     private static final PacManSprites SPRITE_STORE = new PacManSprites();
22     1 usage
23     private PlayerFactory Factory = new PlayerFactory(SPRITE_STORE);
24     6 usages
25     private Player ThePlayer = Factory.createPacMan();
26     3 usages
27     private DefaultPointCalculator calculator = new DefaultPointCalculator();
28     no usages
29     PlayerCollisions collisions = new PlayerCollisions(calculator);
30     1 usage
31     Direction up = Direction.valueOf( name: "NORTH");
32
33     no usages new *
34     @Test
35     void testDefaultPointCalculator(){
36         int oldscore = ThePlayer.getScore(); //gets initial score
37         calculator.pacmanMoved(ThePlayer, up); //player moves
38         assertTrue( actual: ThePlayer.getScore() == oldscore); //player shouldn't be awarded for moving
39         oldscore = ThePlayer.getScore(); // saves current score
40         calculator.consumedAPellet(ThePlayer, pellet); //player eats pellet
41         assertTrue( actual: oldscore < ThePlayer.getScore()); //player should be awarded for eating pellet
42     }
43 }
```

Coverage after implementing method testing for pacmanMoved, consumedAPellet, getScore.

nl	21% (24/110)	12% (80/624)	10% (234/2326)
tudelft	21% (24/110)	12% (80/624)	10% (234/2326)
jpacman	21% (24/110)	12% (80/624)	10% (234/2326)
board	20% (4/20)	13% (14/106)	12% (36/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	14% (22/156)	7% (54/716)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	50% (2/4)	28% (4/14)	19% (8/42)
sprite	83% (10/12)	44% (40/90)	52% (136/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)
PacmanConfigurationE	0% (0/1)	0% (0/2)	0% (0/4)

### Test 3 :

Code for method testing setAlive and is Alive.

Method in:src/main/java/nl/tudelft/jpacman/ui/Player.java

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

import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;

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

no usages new *
public class PlayerTest {

    1 usage
    private static final PacManSprites SPRITE_STORE = new PacManSprites();
    1 usage
    private PlayerFactory Factory = new PlayerFactory(SPRITE_STORE);
    5 usages
    private Player ThePlayer = Factory.createPacMan();

    no usages new *
    @Test
    void testAlive(){
        assertThat(ThePlayer.isAlive()).isEqualTo( expected: true);
    }





















    no usages new *
    @Test
    void testDead(){
        ThePlayer.setAlive(true); //player lives
        assertThat(ThePlayer.isAlive()).isEqualTo( expected: true); //confirms living
        ThePlayer.setAlive(false); //player dies
        assertThat(ThePlayer.isAlive()).isEqualTo( expected: false); //confirms dead
    }
}
```

Coverage after method testing setAlive and is Alive.

nl	21% (24/110)	13% (84/624)	11% (256/2326)
tudelft	21% (24/110)	13% (84/624)	11% (256/2326)
jpacman	21% (24/110)	13% (84/624)	11% (256/2326)
board	20% (4/20)	13% (14/106)	12% (36/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	15% (24/156)	9% (68/716)
npc	0% (0/20)	0% (0/94)	0% (0/474)
points	50% (2/4)	28% (4/14)	19% (8/42)
sprite	83% (10/12)	46% (42/90)	55% (144/260)
ui	0% (0/12)	0% (0/62)	0% (0/254)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)
LauncherSr	0% (0/1)	0% (0/4)	0% (0/29)
PacmanCor	0% (0/1)	0% (0/2)	0% (0/4)

### Task 3:

#### jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
nl.tudelft.jpacman.level		68%		58%	72 155	102 344	21 69	4 12
nl.tudelft.jpacman.npc.ghost		71%		55%	56 105	43 181	5 34	0 8
nl.tudelft.jpacman.ui		77%		47%	54 86	21 144	7 31	0 6
default		0%		0%	12 12	21 21	5 5	1 1
nl.tudelft.jpacman.board		86%		59%	43 93	2 110	0 40	0 7
nl.tudelft.jpacman.sprite		86%		59%	30 70	11 113	5 38	0 5
nl.tudelft.jpacman		69%		25%	12 30	18 52	6 24	1 2
nl.tudelft.jpacman.points		60%		75%	1 11	5 21	0 9	0 2
nl.tudelft.jpacman.game		87%		60%	10 24	4 45	2 14	0 3
nl.tudelft.jpacman.npc		100%		n/a	0 4	0 8	0 4	0 1
Total	1,206 of 4,694	74%	290 of 637	54%	290 590	227 1,039	51 268	6 47

The Results were not similar, in Intellej I got an overall coverage of 21% Whereas in JaCoCo I had 74%

I found the visualizations to be very useful, it was easy to see where tests were missing, and it was nice to have a visualization to catch my eyes and know I don't need to look at green etc.

I would prefer IntelliJ's coverage more, simply because for me it was easier to read the labels and see more useful information faster. JaCoCo had a lot of information that I just wasn't sure what it represented, and I found some of the titles to be misleading. It would be nice if IntelliJ's Class%, Method%, and Line% were done with JaCoCo's style