Maylynn Dimalanta
CS 472 – 1001

JaCoCo Report

Are the coverage results from `JaCoCo` similar to the ones you got from `IntelliJ` in the last task? Why so or why not?

No, I didn't find the coverage results to be similar as the Intellij. This could be due to Intellj only accounting for methods that are called within the unit tests.

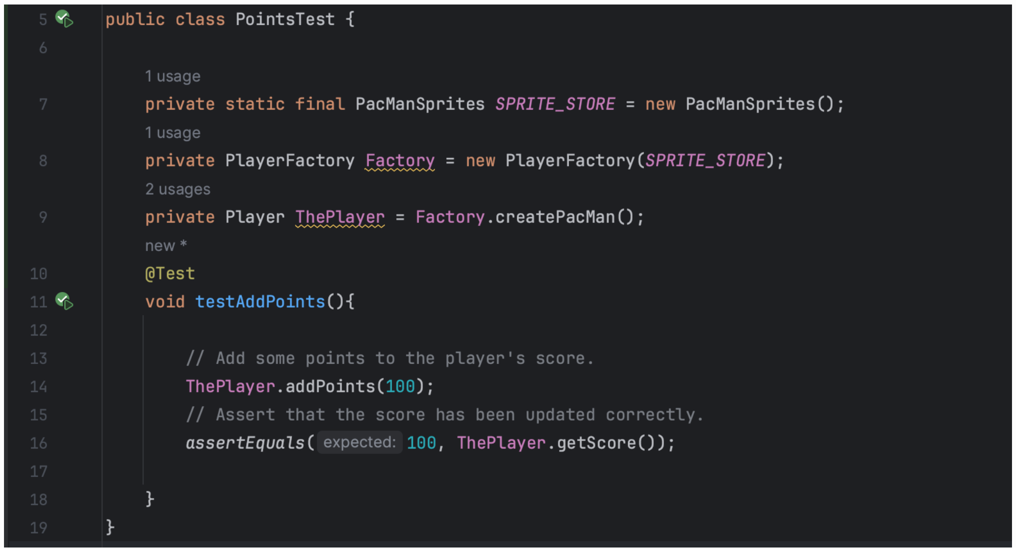
Did you find helpful the source code visualization from JaCoCo on uncovered branches?

Yes, I did find the jacoco source code as a helpful visualization. I liked how it was easy to navigate and readable. I found the percentage scales and additional details about the lines, missed, and classes useful.

Which visualization did you prefer and why? IntelliJ's coverage window or JaCoCo's report?

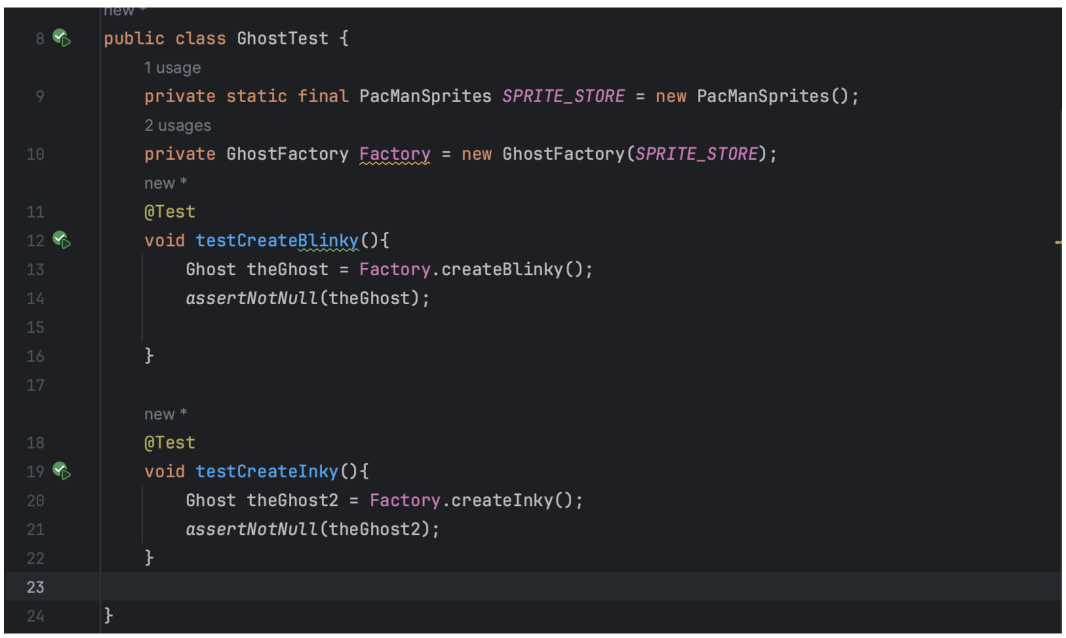
I think I prefer the JaCoCo's report just so that I can see the number information about the missed branches, and I can look at the percentage scales. I do like how the Intellij tells you how many methods are used based on the total though.

addPoints()



```
public class PointsTest {

    1 usage
    private static final PacManSprites SPRITE_STORE = new PacManSprites();
    1 usage
    private PlayerFactory Factory = new PlayerFactory(SPRITE_STORE);
    2 usages
    private Player ThePlayer = Factory.createPacMan();
    new *
    @Test
    void testAddPoints(){

        // Add some points to the player's score.
        ThePlayer.addPoints(100);
        // Assert that the score has been updated correctly.
        assertEquals( expected: 100, ThePlayer.getScore());

    }
}
```

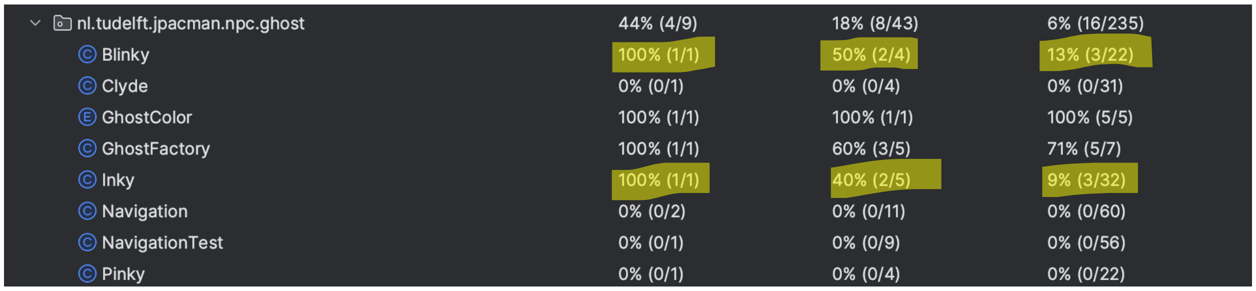
CreateBlinky() & CreateBlinky()



```
new *
public class GhostTest {
    1 usage
    private static final PacManSprites SPRITE_STORE = new PacManSprites();
    2 usages
    private GhostFactory Factory = new GhostFactory(SPRITE_STORE);
    new *
    @Test
    void testCreateBlinky(){
        Ghost theGhost = Factory.createBlinky();
        assertNotNull(theGhost);

    }

    new *
    @Test
    void testCreateInky(){
        Ghost theGhost2 = Factory.createInky();
        assertNotNull(theGhost2);
    }

}
```

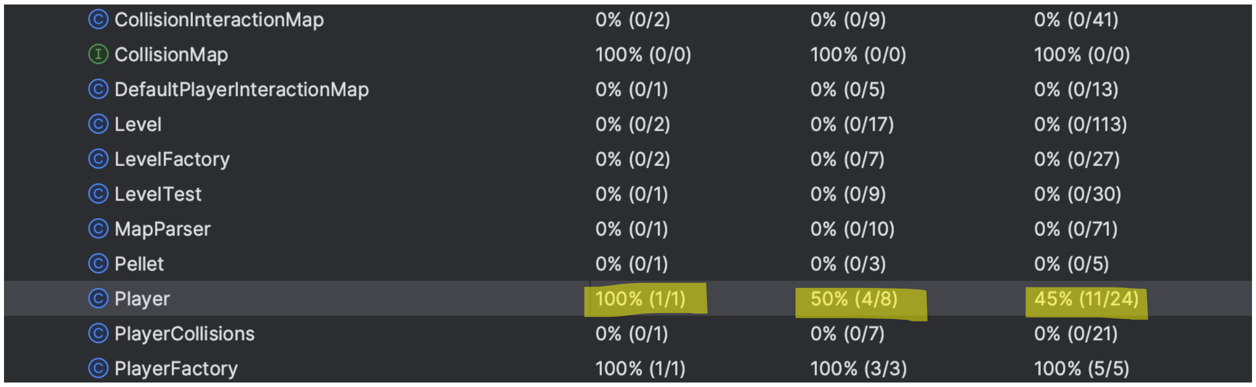
Coverage Results

After CreateBlinky() and CreateInky()



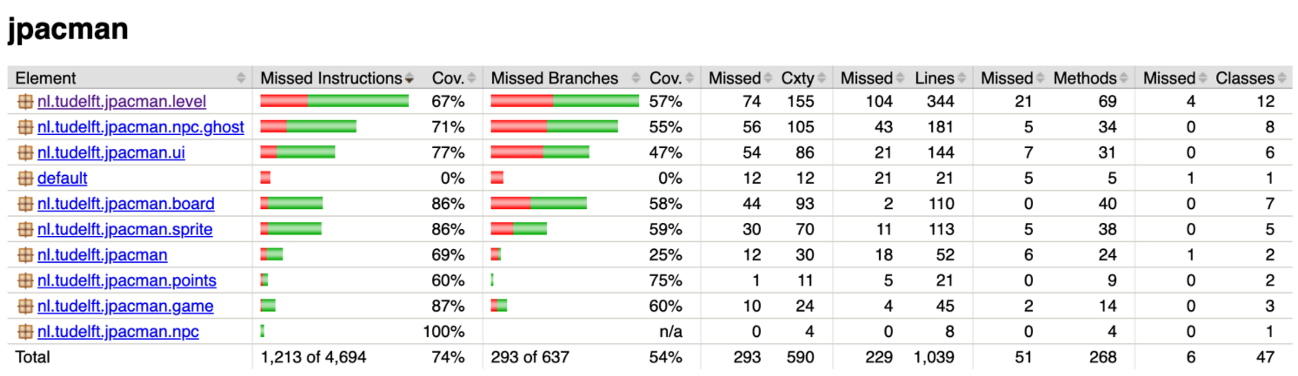
nl.tudelft.jpacman.npc.ghost	44% (4/9)	18% (8/43)	6% (16/235)
Blinky	100% (1/1)	50% (2/4)	13% (3/22)
Clyde	0% (0/1)	0% (0/4)	0% (0/31)
GhostColor	100% (1/1)	100% (1/1)	100% (5/5)
GhostFactory	100% (1/1)	60% (3/5)	71% (5/7)
Inky	100% (1/1)	40% (2/5)	9% (3/32)
Navigation	0% (0/2)	0% (0/11)	0% (0/60)
NavigationTest	0% (0/1)	0% (0/9)	0% (0/56)
Pinky	0% (0/1)	0% (0/4)	0% (0/22)

After addPoints()



CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/41)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)
Level	0% (0/2)	0% (0/17)	0% (0/113)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)
Player	100% (1/1)	50% (4/8)	45% (11/24)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)

JaCoCo



jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
nl.tudelft.jpacman.level	🟥🟩	67%	🟥🟩	57%	74	155	104	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%	🟥🟩	58%	44	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	1,213 of 4,694	74%	293 of 637	54%	293	590	229	1,039	51	268	6	47