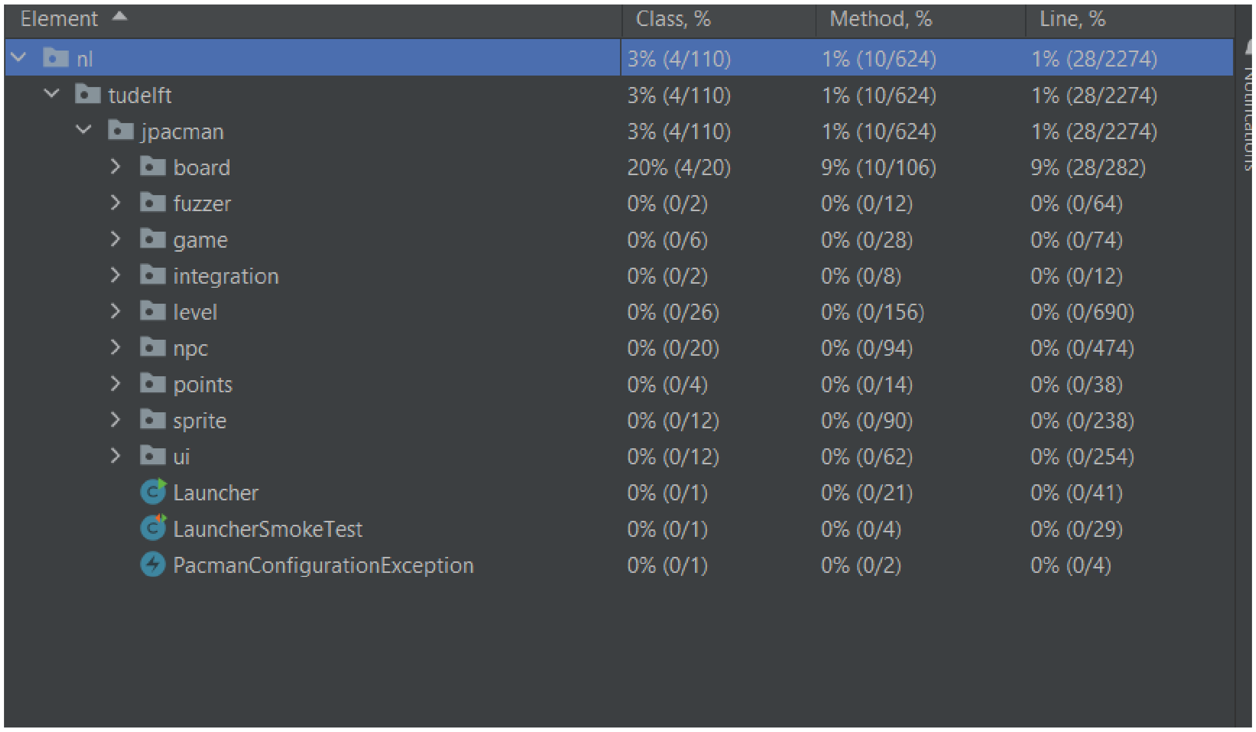
Cody Nguyen
CS 472
Testing Lab
Fork Repository: https://github.com/Proxhi/472-2023-G3

Task 1

<u>Initial Coverage</u>

This coverage report shows that 3% class coverage is being covered with the given tests that came with the source.

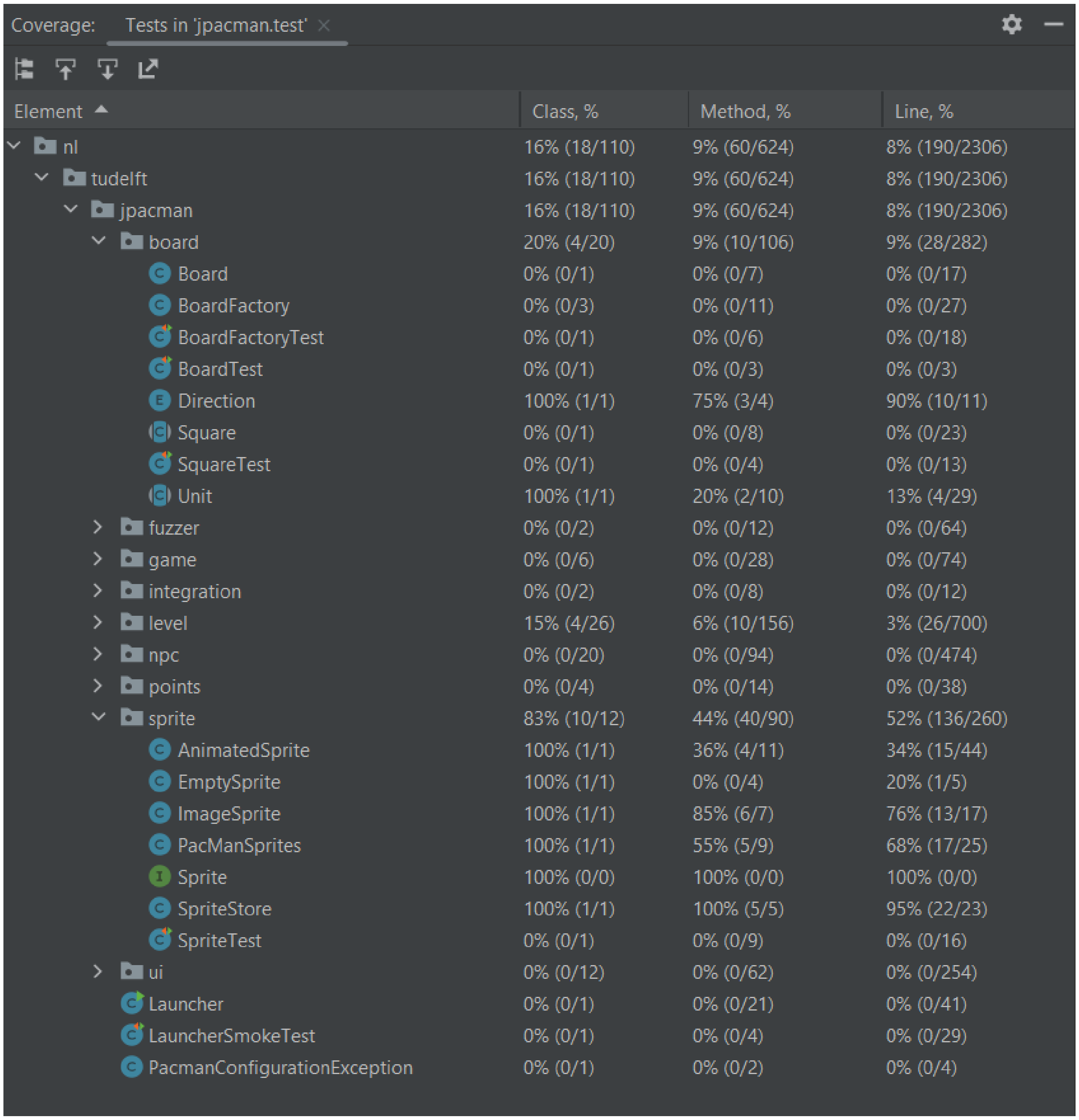


Element	Class, %	Method, %	Line, %
nl	3% (4/110)	1% (10/624)	1% (28/2274)
tudelft	3% (4/110)	1% (10/624)	1% (28/2274)
jpacman	3% (4/110)	1% (10/624)	1% (28/2274)
board	20% (4/20)	9% (10/106)	9% (28/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	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)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

Task 2

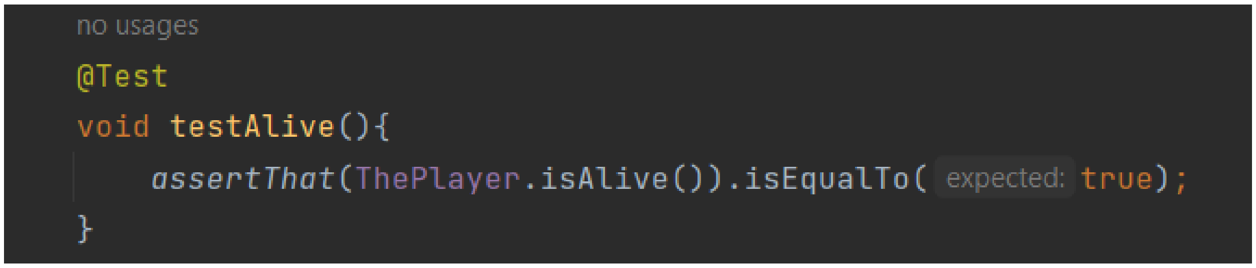
<u>Adding isAlive()</u>

isAlive() is a method which checks if Pacman is alive. Adding this test increased the class coverage to 16% and the method coverage to 9%.



Coverage: Tests in 'jpacman.test'

Element	Class, %	Method, %	Line, %
nl	16% (18/110)	9% (60/624)	8% (190/2306)
tudelft	16% (18/110)	9% (60/624)	8% (190/2306)
jpacman	16% (18/110)	9% (60/624)	8% (190/2306)
board	20% (4/20)	9% (10/106)	9% (28/282)
Board	0% (0/1)	0% (0/7)	0% (0/17)
BoardFactory	0% (0/3)	0% (0/11)	0% (0/27)
BoardFactoryTest	0% (0/1)	0% (0/6)	0% (0/18)
BoardTest	0% (0/1)	0% (0/3)	0% (0/3)
Direction	100% (1/1)	75% (3/4)	90% (10/11)
Square	0% (0/1)	0% (0/8)	0% (0/23)
SquareTest	0% (0/1)	0% (0/4)	0% (0/13)
Unit	100% (1/1)	20% (2/10)	13% (4/29)
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% (10/12)	44% (40/90)	52% (136/260)
AnimatedSprite	100% (1/1)	36% (4/11)	34% (15/44)
EmptySprite	100% (1/1)	0% (0/4)	20% (1/5)
ImageSprite	100% (1/1)	85% (6/7)	76% (13/17)
PacManSprites	100% (1/1)	55% (5/9)	68% (17/25)
Sprite	100% (0/0)	100% (0/0)	100% (0/0)
SpriteStore	100% (1/1)	100% (5/5)	95% (22/23)
SpriteTest	0% (0/1)	0% (0/9)	0% (0/16)
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)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)



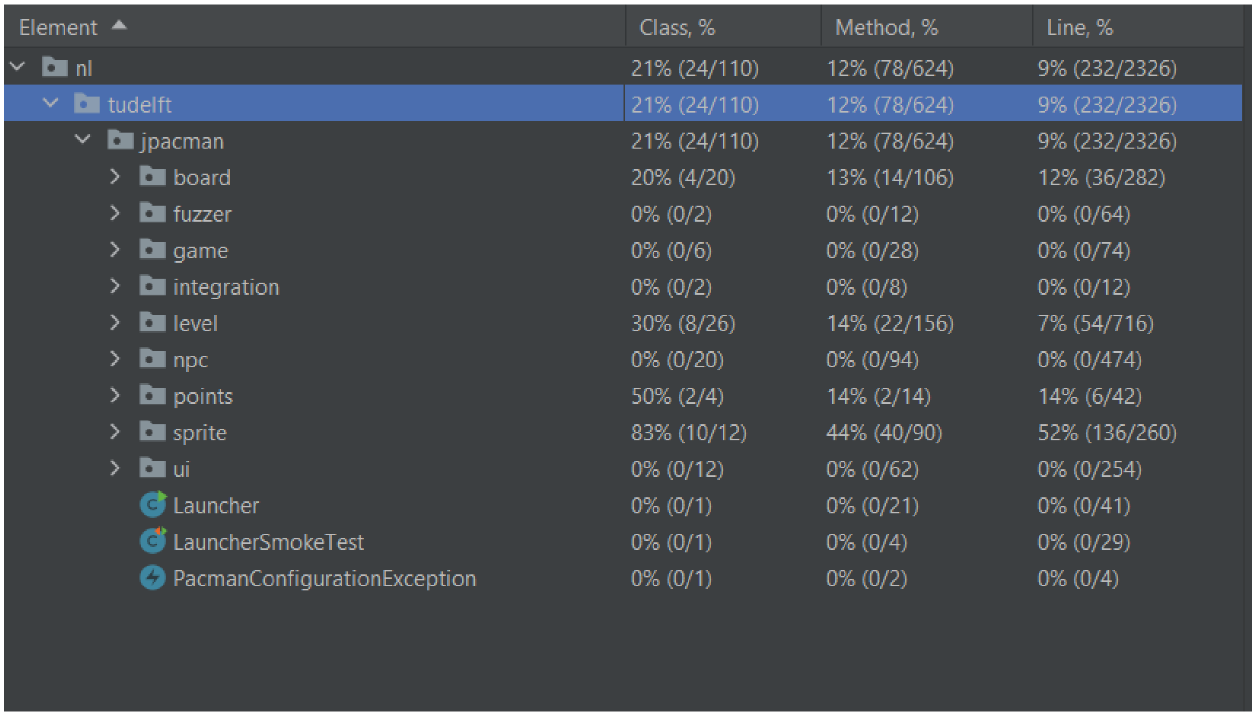
```
no usages
@Test
void testAlive(){
    assertThat(ThePlayer.isAlive()).isEqualTo( expected: true);
}
```

Task 2.1

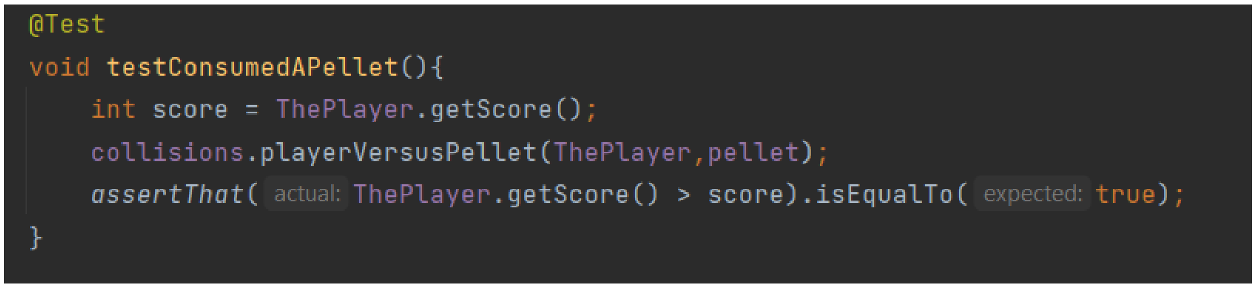
Adding testConsumedAPellet()

Method tested: src/main/java/nl/tudelft/jpacman/points/PointCalculator.java (consumedAPellet)

This test gets the initial score of the player before it consumes a pellet and compares the score after with the initial. Adding this test has increased our class coverage to 21% and our method coverage to 12%.

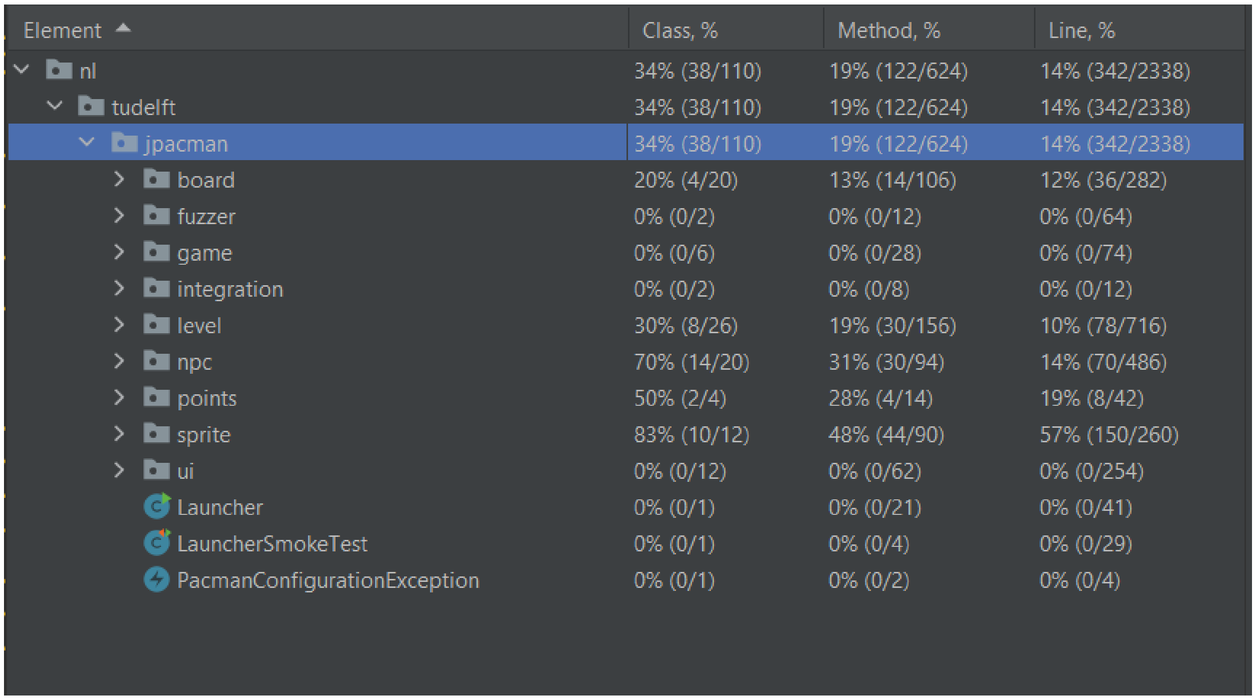


Element	Class, %	Method, %	Line, %
nl	21% (24/110)	12% (78/624)	9% (232/2326)
tudelft	21% (24/110)	12% (78/624)	9% (232/2326)
jpacman	21% (24/110)	12% (78/624)	9% (232/2326)
board	20% (4/20)	13% (14/106)	12% (36/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (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)	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)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)

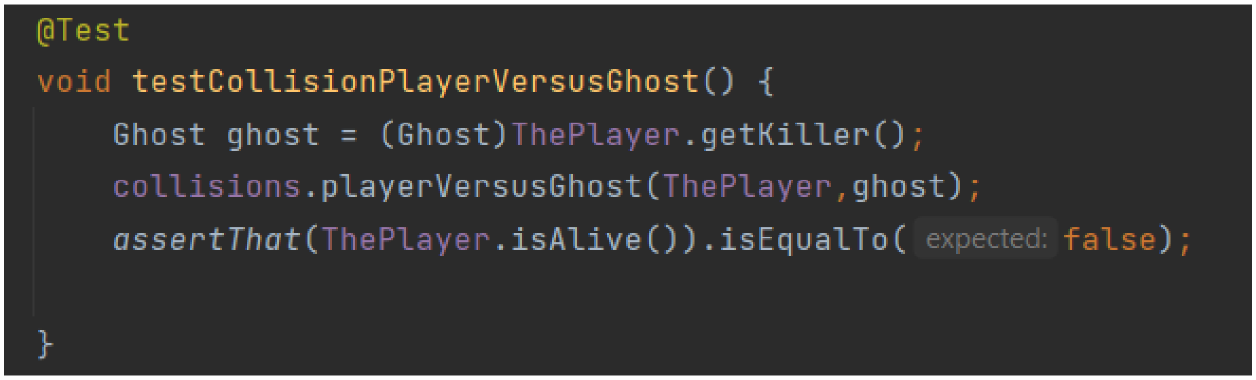


```
@Test
void testConsumedAPellet(){
    int score = ThePlayer.getScore();
    collisions.playerVersusPellet(ThePlayer,pellet);
    assertThat( actual: ThePlayer.getScore() > score).isEqualTo( expected: true);
}
```

Adding testCollisionPlayerVersusGhost()
Method tested: src/main/java/nl/tudelft/jpacman/level/PlayerCollisions.java (playerVersusGhost)
This test grabs the killer of the player, which is a ghost, and checks to see if the player is not alive after the collision. This test brought the class coverage to 34% and the method coverage to 19%.



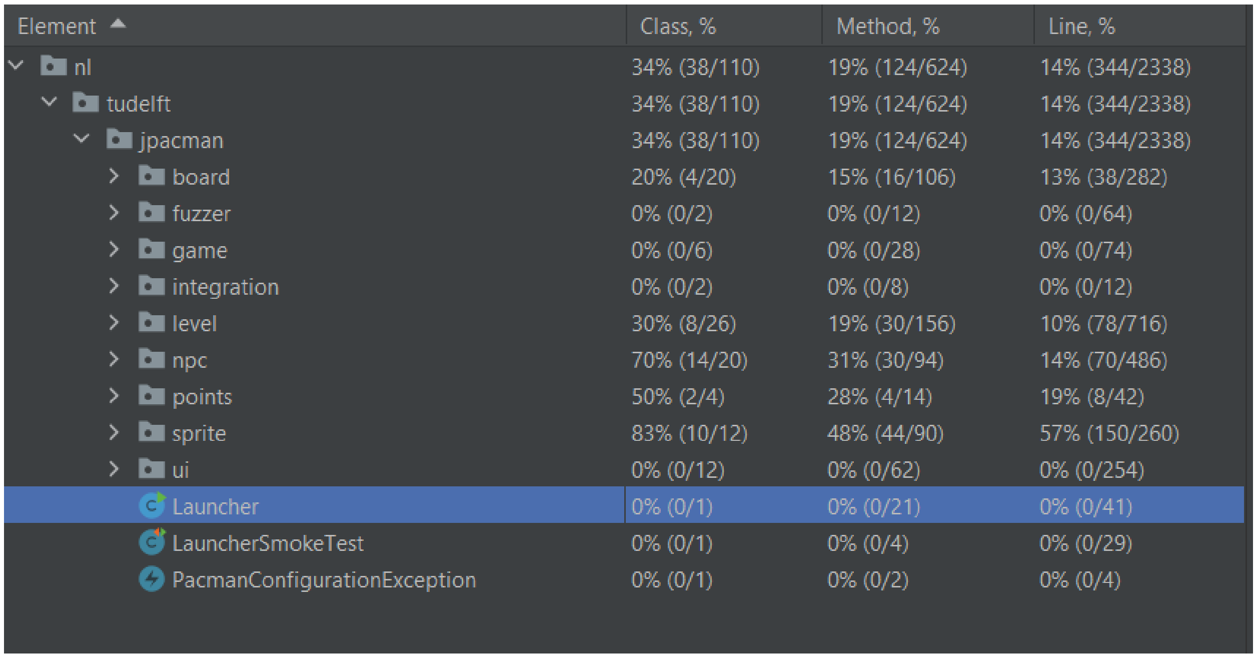
Element	Class, %	Method, %	Line, %
nl	34% (38/110)	19% (122/624)	14% (342/2338)
tudelft	34% (38/110)	19% (122/624)	14% (342/2338)
jpacman	34% (38/110)	19% (122/624)	14% (342/2338)
board	20% (4/20)	13% (14/106)	12% (36/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	19% (30/156)	10% (78/716)
npc	70% (14/20)	31% (30/94)	14% (70/486)
points	50% (2/4)	28% (4/14)	19% (8/42)
sprite	83% (10/12)	48% (44/90)	57% (150/260)
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)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)



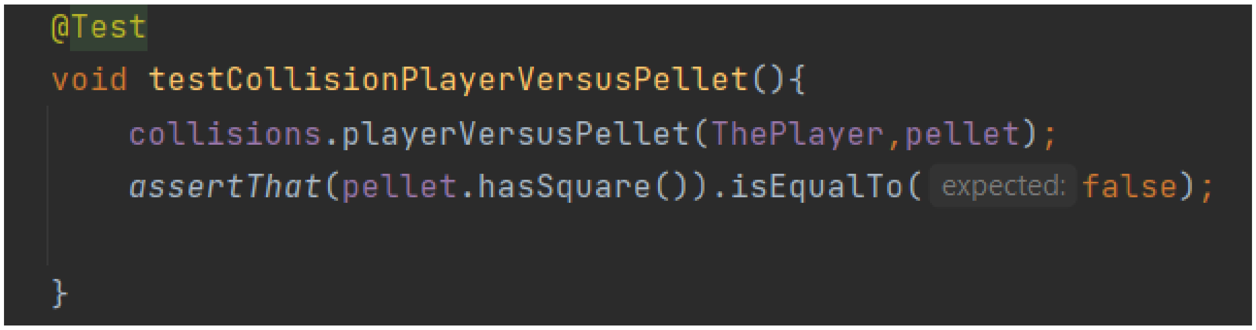
```java
@Test
void testCollisionPlayerVersusGhost() {
    Ghost ghost = (Ghost)ThePlayer.getKiller();
    collisions.playerVersusGhost(ThePlayer,ghost);
    assertThat(ThePlayer.isAlive()).isEqualTo( expected: false);

}
```

Adding testCollisionPlayerVersusPellet()

Method tested: src/main/java/nl/tudelft/jpacman/level/PlayerCollisions.java (playerVersusPellet)
This test checks to see if the pellet has disappeared from the spot it was located after Pacman consumed it. This kept the same coverage as the previous test.

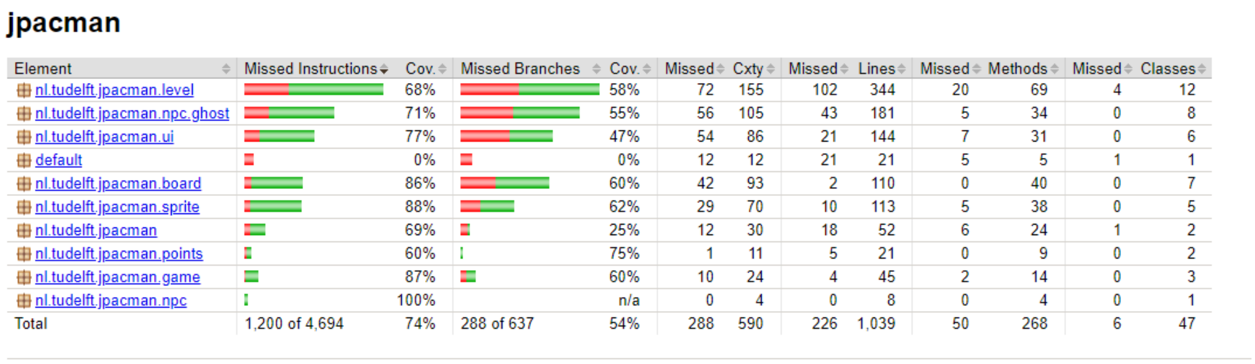


Element	Class, %	Method, %	Line, %
nl	34% (38/110)	19% (124/624)	14% (344/2338)
tudelft	34% (38/110)	19% (124/624)	14% (344/2338)
jpacman	34% (38/110)	19% (124/624)	14% (344/2338)
board	20% (4/20)	15% (16/106)	13% (38/282)
fuzzer	0% (0/2)	0% (0/12)	0% (0/64)
game	0% (0/6)	0% (0/28)	0% (0/74)
integration	0% (0/2)	0% (0/8)	0% (0/12)
level	30% (8/26)	19% (30/156)	10% (78/716)
npc	70% (14/20)	31% (30/94)	14% (70/486)
points	50% (2/4)	28% (4/14)	19% (8/42)
sprite	83% (10/12)	48% (44/90)	57% (150/260)
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)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)



```
@Test
void testCollisionPlayerVersusPellet(){
    collisions.playerVersusPellet(ThePlayer,pellet);
    assertThat(pellet.hasSquare()).isEqualTo( expected: false);

}
```

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	20	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%		60%	42	93	2	110	0	40	0	7
nl.tudelft.jpacman.sprite		88%		62%	29	70	10	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	1,200 of 4,694	74%	288 of 637	54%	288	590	226	1,039	50	268	6	47

- The results weren’t similar to the IntelliJ results, as JaCoCo shows a 74% coverage compared to a 34% coverage. There is the possibility that JaCoCo is looking at different aspects of the test to determine coverbility.
- JaCoCo provides a good visualization in the part they include colored bar graphs; however, I don’t like how the table is formatted. The format makes the information harder to read and determine the statistics of the coverage.
- I personally prefer the Intellij visualization as it’s easier to understand and read. Intellij provides a better formatted drop down graph, compared to the table that shows everything. With Intellij, I’m able to filter the statistics I want to see.