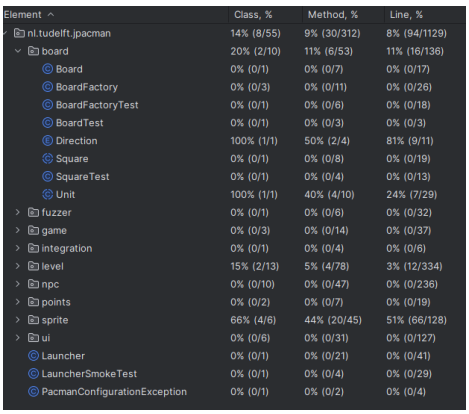
Tristan Ferguson

Link to forked repository: https://github.com/fergut3/jpacman

Task 2 and 2.1:

Coverage reports after creating tests:

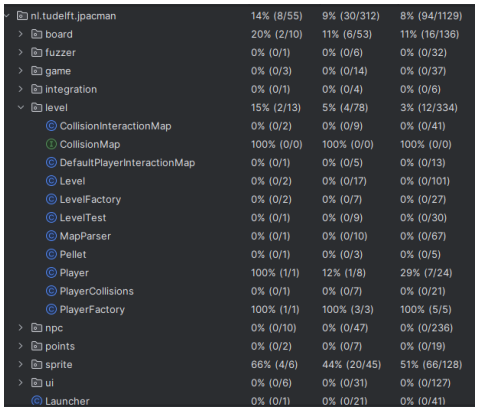
Testing setDirection() and getDirection() do not seem to be very hard because I was able to create a player class and then use that to set the direction to the current direction it has. Doing this covered for both tests efficiently. After creating tests for SetDirection() and GetDirection(): Unit increased from 13% to 24%



Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	14% (8/55)	9% (30/312)	8% (94/1129)
board	20% (2/10)	11% (6/53)	11% (16/136)
Board	0% (0/1)	0% (0/7)	0% (0/17)
BoardFactory	0% (0/3)	0% (0/11)	0% (0/26)
BoardFactoryTest	0% (0/1)	0% (0/6)	0% (0/18)
BoardTest	0% (0/1)	0% (0/3)	0% (0/3)
Direction	100% (1/1)	50% (2/4)	81% (9/11)
Square	0% (0/1)	0% (0/8)	0% (0/19)
SquareTest	0% (0/1)	0% (0/4)	0% (0/13)
Unit	100% (1/1)	40% (4/10)	24% (7/29)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)
game	0% (0/3)	0% (0/14)	0% (0/37)
integration	0% (0/1)	0% (0/4)	0% (0/6)
level	15% (2/13)	5% (4/78)	3% (12/334)
npc	0% (0/10)	0% (0/47)	0% (0/236)
points	0% (0/2)	0% (0/7)	0% (0/19)
sprite	66% (4/6)	44% (20/45)	51% (66/128)
ui	0% (0/6)	0% (0/31)	0% (0/127)
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)

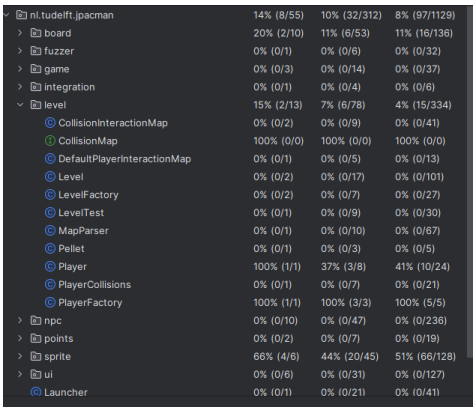
```
@Test
void DirectionsTest(){
    ThePlayer.setDirection(ThePlayer.getDirection());
}
```

Before getSprite() test:



nl.tudelft.jpacman	14% (8/55)	9% (30/312)	8% (94/1129)
board	20% (2/10)	11% (6/53)	11% (16/136)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)
game	0% (0/3)	0% (0/14)	0% (0/37)
integration	0% (0/1)	0% (0/4)	0% (0/6)
level	15% (2/13)	5% (4/78)	3% (12/334)
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% (0/101)
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% (0/67)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)
Player	100% (1/1)	12% (1/8)	29% (7/24)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)
npc	0% (0/10)	0% (0/47)	0% (0/236)
points	0% (0/2)	0% (0/7)	0% (0/19)
sprite	66% (4/6)	44% (20/45)	51% (66/128)
ui	0% (0/6)	0% (0/31)	0% (0/127)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)

After getSprite() test:



nl.tudelft.jpacman	14% (8/55)	10% (32/312)	8% (97/1129)
board	20% (2/10)	11% (6/53)	11% (16/136)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)
game	0% (0/3)	0% (0/14)	0% (0/37)
integration	0% (0/1)	0% (0/4)	0% (0/6)
level	15% (2/13)	7% (6/78)	4% (15/334)
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% (0/101)
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% (0/67)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)
Player	100% (1/1)	37% (3/8)	41% (10/24)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)
npc	0% (0/10)	0% (0/47)	0% (0/236)
points	0% (0/2)	0% (0/7)	0% (0/19)
sprite	66% (4/6)	44% (20/45)	51% (66/128)
ui	0% (0/6)	0% (0/31)	0% (0/127)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)

The Unit.getSprite method was just as simple as getDirection and setDirection. By setting the command, it would error out if the sprite set to it was not a valid sprite. After creating the test command, player line coverage increased from 29% to 41%

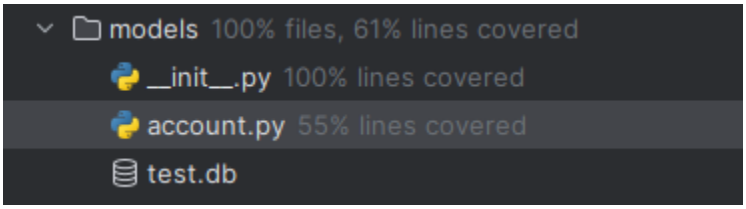
```
@Test
void TestSprite(){
    ThePlayer.getSprite();
}
```

Task 3:

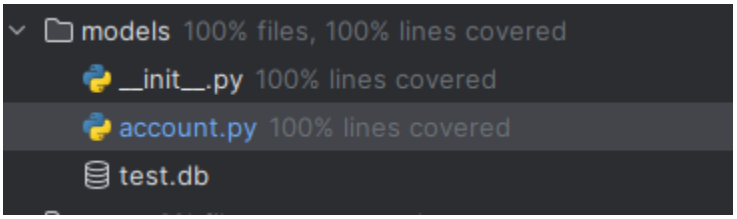
JaCoco's reporting seems to be similar to the reporting that was done in IntelliJ since I was able to locate the player method coverage and was able to see that IsAlive() was at 100%. This makes sense since a test was made for that method specifically. JaCoCo's detail in reporting the test coverage is very useful. I prefer this over IntelliJ because of the ability to easily see which parts of source code are covered since it is highlighted in either green or red.

Task 4:

Before creating extra test functions



After creating extra test functions:



Functions that were made for tests in test_account.py:

test__repr(self)

test_from_dict(self)

test_to_dict(self)

test_create(self)

test_update(self)

test_delete(self)

test_find(self)