Task 2.1

I added unit tests for 3 methods, /level/MapParser.getBoardCreator, /level/Player.addPoints, and /game/GameFactory.getPlayerFactory.

Here is the code for the /level/MapParser.getBoardCreator method test:

GameFactoryTest.java | BoardCreatorTest.java

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

import nl.tudelft.jpacman.board.BoardFactory;
import nl.tudelft.jpacman.npc.ghost.GhostFactory;
import nl.tudelft.jpacman.points.DefaultPointCalculator;
import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;
import static org.assertj.core.api.Assertions.assertThat;

/**
 * Test if board is created properly
 *
 * @author Dillon Davidson
 */
public class BoardCreatorTest
{
    3 usages
    private static final PacManSprites someSprites = new PacManSprites();
    1 usage
    private static final GhostFactory someGhostFactory = new GhostFactory(someSprites);
    1 usage
    private static final DefaultPointCalculator someCalculator = new DefaultPointCalculator();
    1 usage
    private LevelFactory someLevelFactory = new LevelFactory(someSprites, someGhostFactory, someCalculator);
    2 usages
    private BoardFactory someBoardFactory = new BoardFactory(someSprites);
    1 usage
    private MapParser someMapParser = new MapParser(someLevelFactory, someBoardFactory);

    @Test
    void testGetBoardCreator() { assertThat(someMapParser.getBoardCreator()).isEqualTo(someBoardFactory); }
}
```

Here is the code for the /level/Player.addPoints method test:

AddPointsTest.java

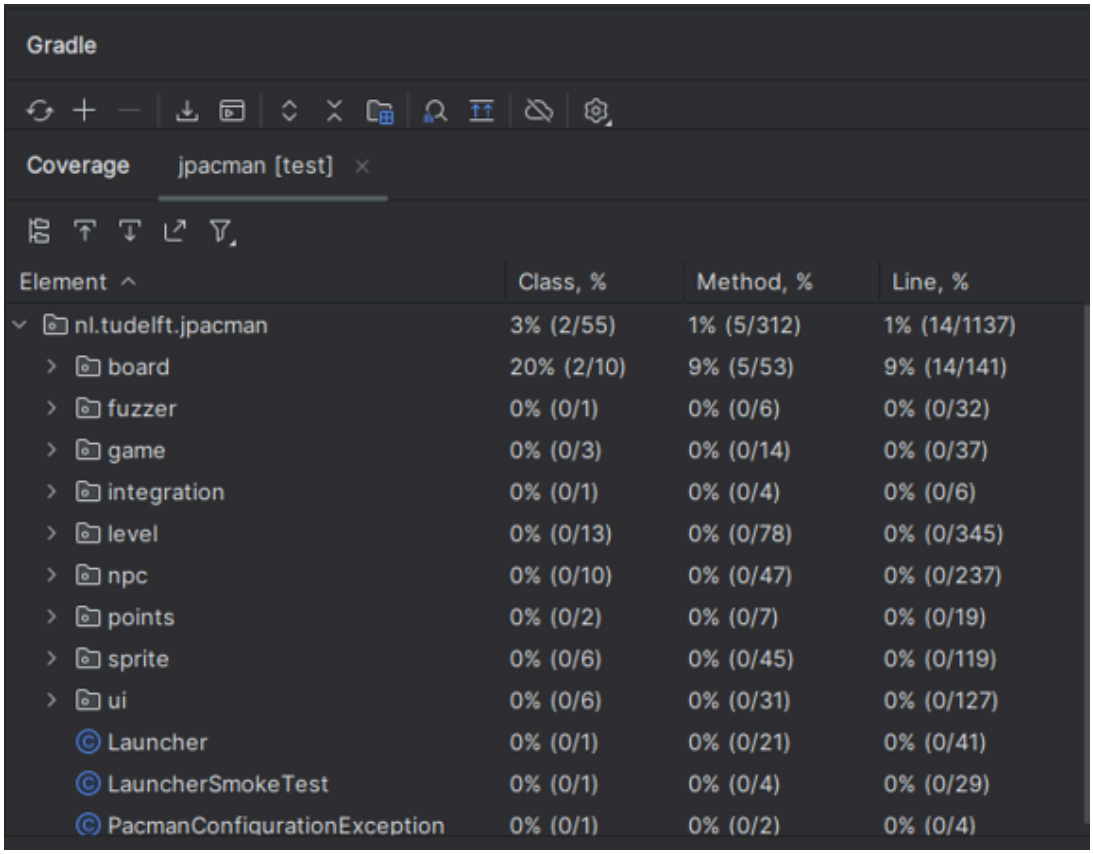
```java
package nl.tudelft.jpacman.level;
import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;

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

/**
 * Test if points get added
 *
 * @author Dillon Davidson
 */

public class AddPointsTest
{

    private static final PacManSprites someSprites = new PacManSprites();

    private PlayerFactory someFactory = new PlayerFactory(someSprites);

    private Player somePlayer = someFactory.createPacMan();


    @Test
    void testAddPoints()
    {
        somePlayer.addPoints(1);
        assertThat(somePlayer.getScore()).isEqualTo( expected: 1);
    }
}

```

Here is the code for the /game/GameFactory.getPlayerFactory method test:

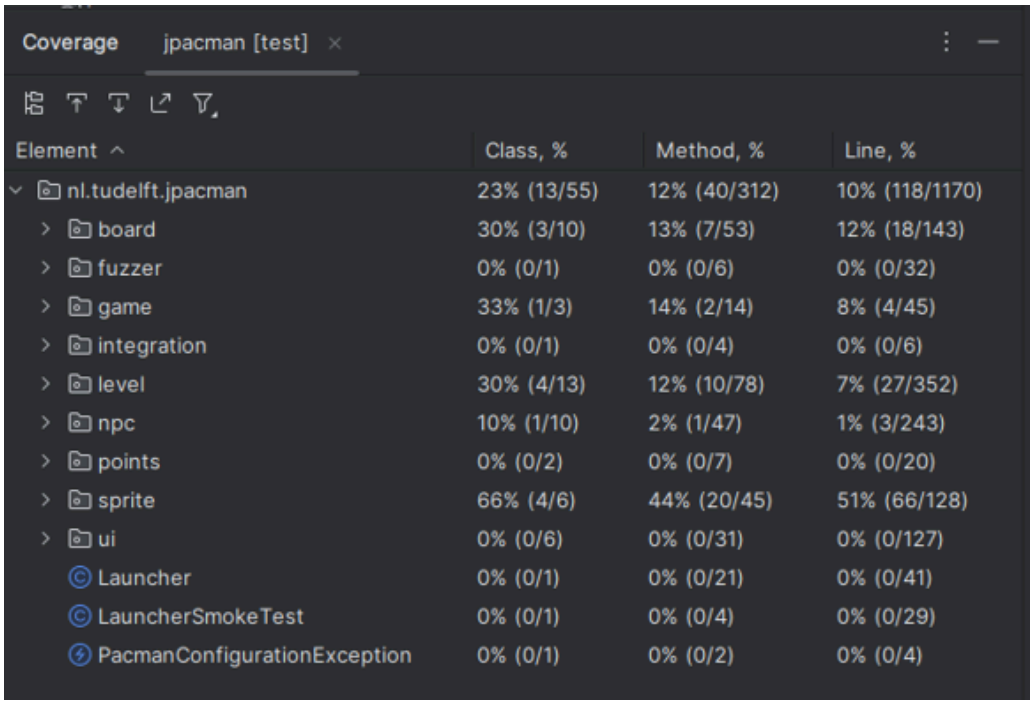
GameFactoryTest.java | BoardCreatorTest.java

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

import nl.tudelft.jpacman.level.PlayerFactory;
import nl.tudelft.jpacman.sprite.PacManSprites;
import org.junit.jupiter.api.Test;
import static org.assertj.core.api.Assertions.assertThat;

/**
 * Test if game factory is returned
 *
 * @author Dillon Davidson
 */
public class GameFactoryTest
{
    1 usage
    private static final PacManSprites someSprites = new PacManSprites();
    2 usages
    private PlayerFactory someFactory = new PlayerFactory(someSprites);
    1 usage
    private GameFactory someGameFactory = new GameFactory(someFactory);

    @Test
    void testGetPlayerFactory()
    {
        assertThat(someGameFactory.getPlayerFactory()).isEqualTo(someFactory);
    }
}

```

Here was the coverage at the beginning before adding my tests:



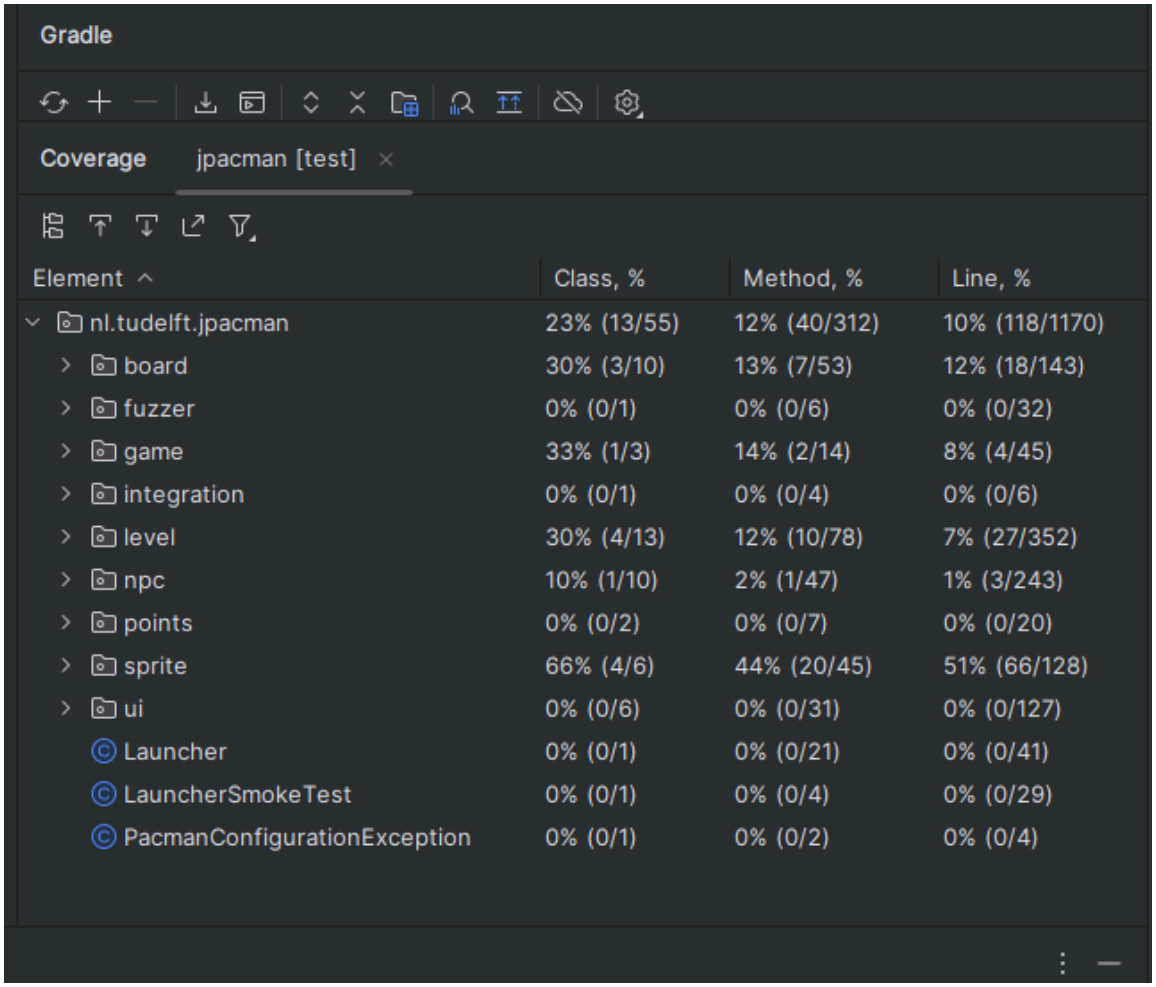
And here was the coverage after adding all my tests:



Making the tests was not very difficult. The hardest part was correctly generating some of the class objects correctly to get all of the constructors to work.

Task 3

Here is the output from IntelliJ after adding my 3 tests



And here is the output from the JaCoCo report from the index.html.

jpacman

jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
nl.tudelft.jpacman.ui		77%		47%	54	86	21	144	7	31	0	6
nl.tudelft.jpacman.sprite		86%		59%	30	70	11	113	5	38	0	5
nl.tudelft.jpacman.points		60%		75%	1	11	5	21	0	9	0	2
nl.tudelft.jpacman.npc.ghost		71%		55%	56	105	43	181	5	34	0	8
nl.tudelft.jpacman.npc		100%		n/a	0	4	0	8	0	4	0	1
nl.tudelft.jpacman.level		67%		57%	73	155	103	344	20	69	4	12
nl.tudelft.jpacman.game		89%		60%	9	24	3	45	1	14	0	3
nl.tudelft.jpacman.board		86%		58%	44	93	2	110	0	40	0	7
nl.tudelft.jpacman		69%		25%	12	30	18	52	6	24	1	2
default		0%		0%	12	12	21	21	5	5	1	1
Total	1,207 of 4,694	74%	293 of 637	54%	291	590	227	1,039	49	268	6	47

1. I think these results are pretty similar. Before adding my tests in my screenshot in Task 2.1, the game package had no coverage, and now it's got pretty much full coverage, specifically on the gameFactory class and game class.
2. JaCoCo's visualization is very helpful on the uncovered branches. This is an amazing way to view branches.
3. I like that JaCoCo's report is interactive. Being able to see literally what lines are covered and are not covered is amazing. It is extremely easy to use and seems very good for development.

Task 4

Here are my code snippets:

```
def test_from_dict(self):
    """Test account from dict"""
    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)
    someDict = {
        "name": "Testing"
    }
    account.from_dict(someDict)
    self.assertEqual(account.name, "Testing")
```

```python
def test_update(self):
    """Test account update"""
    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)

    # Testing with blank id
    try:
        account.update()
    except DataValidationError:
        self.assertEqual(account.id, None)

    # Testing with dummy id
    account.id = 5
    account.update()
    self.assertEqual(account.id, 5)
```

```python
def test_delete(self):
    """Test account deletion"""
    size = len(ACCOUNT_DATA)
    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)
    account.create()
    account.delete()
    self.assertEqual(len(ACCOUNT_DATA), size)
```

```python
def test_find(self):
    """Test account finding"""
    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)
    # Actually create the account (this took me a while)
    account.create()

    # Find the account by its ID
    found_account = Account.find(account.id)

    # Assert that the found account matches the original account
    self.assertIsNotNone(found_account)
    self.assertEqual(account.id, found_account.id)
    self.assertEqual(account.name, found_account.name)
    self.assertEqual(account.email, found_account.email)
    self.assertEqual(account.phone_number, found_account.phone_number)
    self.assertEqual(account.disabled, found_account.disabled)
    self.assertEqual(account.date_joined, found_account.date_joined)
```

Task 5

Here was the first test I added.

```
def test_update_a_container(self):
    # Make a bad update first
    badCheck = self.client.get('counters/foob')
    self.assertEqual(badCheck.status_code, status.HTTP_404_NOT_FOUND)
    badUpdate = self.client.put('/counters/foob')

    # Make a call to create a Counter
    client = app.test_client()
    result = client.post('/counters/foob')

    # Ensure that it returned a successful return code
    self.assertEqual(result.status_code, status.HTTP_201_CREATED)

    # Check the counter value as a baseline
    initialCounterResponse = self.client.get('/counters/foob')
    initialCounterValue = initialCounterResponse.json['foob']

    # Make a call to Update the counter that you just created
    updateResponse = self.client.put('/counters/foob')

    # Ensure that it returned a successful return code
    self.assertEqual(updateResponse.status_code, status.HTTP_200_OK)

    # Check that the counter value is one more than the baseline in step 3
    updatedCounter = self.client.get('/counters/foob')
    updatedCounterValue = updatedCounter.json['foob']
    self.assertEqual(updatedCounterValue, initialCounterValue + 1)
```

Here's what nosetests outputs with just this:

```
PS C:\CS\CS_472\tdd> nosetests

Counter tests
- It should create a counter
- It should return an error for duplicates
- update a container (FAILED)

======================================================================
FAIL: test_update_a_container (test_counter.CounterTest)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\CS\CS_472\tdd\tests\test_counter.py", line 42, in test_update_a_container
    self.assertEqual(badCheck.status_code, status.HTTP_404_NOT_FOUND)
AssertionError: 405 != 404

Name             Stmts   Miss  Cover   Missing
------------------------------------------------
src\counter.py      12      0   100%
src\status.py        6      0   100%
------------------------------------------------
TOTAL               18      0   100%
----------------------------------------------------------------------
Ran 3 tests in 0.310s

FAILED (failures=1)

PS C:\CS\CS_472\tdd>
```

After writing update_counter and get…

```python
@app.route('/counters/<name>', methods=['PUT'])
def update_counter(name):
    """Update a counter"""
    app.logger.info(f"Request to update counter: {name}")
    global COUNTERS
    if name not in COUNTERS:
        return {"Message": f"Counter {name} does not exist"}, status.HTTP_404_NOT_FOUND
    COUNTERS[name] += 1
    return {name: COUNTERS[name]}, status.HTTP_200_OK
```

Initially nosetests gave me this

```
PS C:\CS\CS_472\tdd> nosetests

Counter tests
- It should create a counter
- It should return an error for duplicates
- update a container

Name             Stmts   Miss  Cover   Missing
----------------------------------------------
src\counter.py      24      2    92%   29, 39
src\status.py        6      0   100%
----------------------------------------------
TOTAL               30      2    93%
----------------------------------------------------------------------
Ran 3 tests in 0.307s

OK
```

Because I forgot to include cases for if the name is not in counters in the tests. I included it in my original screenshot in my test_counter.py function.

After adding these two test cases in… I got full coverage!

```
PS C:\CS\CS_472\tdd> nosetests

Counter tests
- It should create a counter
- It should return an error for duplicates
- update a container

Name             Stmts   Miss  Cover   Missing
----------------------------------------------
src\counter.py      24      0   100%
src\status.py        6      0   100%
----------------------------------------------
TOTAL               30      0   100%
----------------------------------------------------------------------
Ran 3 tests in 0.197s

OK

PS C:\CS\CS_472\tdd>
```