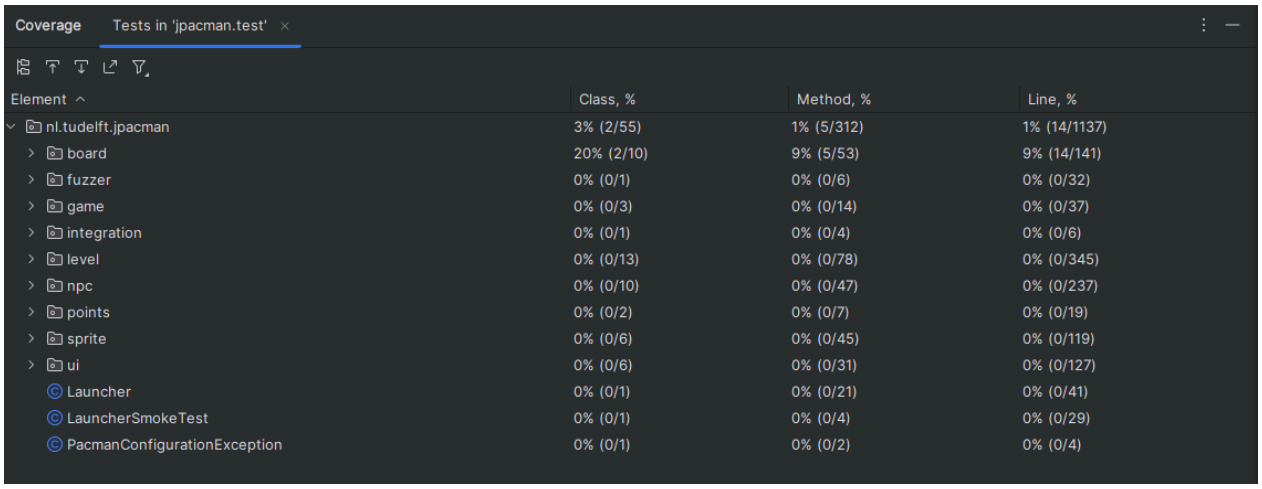
Github fork: https://github.com/SwiftNemesis/tdd

Task 1

Question: Is this coverage good enough?

- Answer: No the coverage is not good enough. As seen in the image below, only about 3% of the classes, 1% of the methods, and 1% of the lines of code are covered during this test. That is inadequate coverage.



Coverage | Tests in 'jpacman.test'

Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	3% (2/55)	1% (5/312)	1% (14/1137)
board	20% (2/10)	9% (5/53)	9% (14/141)
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	0% (0/13)	0% (0/78)	0% (0/345)
npc	0% (0/10)	0% (0/47)	0% (0/237)
points	0% (0/2)	0% (0/7)	0% (0/19)
sprite	0% (0/6)	0% (0/45)	0% (0/119)
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)

Task 2

PlayerTest Code Snippet:

```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;

SwiftNemesis
public class PlayerTest {

        1 usage
        private static final PacManSprites PACMAN_SPRITE = new PacManSprites();
        1 usage
        private PlayerFactory playerFactory = new PlayerFactory(PACMAN_SPRITE);
        1 usage
        private Player ThePlayer = playerFactory.createPacMan();

        SwiftNemesis
        @Test
        void isAlive() { assertThat(ThePlayer.isAlive()).isEqualTo( expected: true); }
}
```

Test Coverage:

Coverage | Tests in 'jpacman'

Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	14% (8/55)	9% (30/312)	8% (93/1151)
board	20% (2/10)	9% (5/53)	9% (14/141)
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)	6% (5/78)	3% (13/350)
npc	0% (0/10)	0% (0/47)	0% (0/237)
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)

Task 2.1:

Code Snippets:

ClydeTest:

```java
package nl.tudelft.jpacman.npc.ghost;

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

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


SwiftNemesis
public class ClydeTest {
        1 usage
        private static final PacManSprites PACMAN_SPRITE = new PacManSprites();
        1 usage
        private static final GhostFactory GHOST_FACTORY = new GhostFactory(PACMAN_SPRITE);

        SwiftNemesis
        @Test
        void doesExist(){
            Ghost clyde = GHOST_FACTORY.createClyde();
            assertThat(clyde).isNotNull();
        }

}
```

PinkyTest:

```java
package nl.tudelft.jpacman.npc.ghost;

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

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


SwiftNemesis
public class PinkyTest {
    1 usage
    private static final PacManSprites PACMAN_SPRITE = new PacManSprites();
    1 usage
    private static final GhostFactory GHOST_FACTORY = new GhostFactory(PACMAN_SPRITE);

    SwiftNemesis
    @Test
    void doesExist(){
        Ghost pinky = GHOST_FACTORY.createPinky();
        assertThat(pinky).isNotNull();
    }

}
```

InkyTest:

```java
package nl.tudelft.jpacman.npc.ghost;

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

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


SwiftNemesis
public class InkyTest {
    1 usage
    private static final PacManSprites PACMAN_SPRITE = new PacManSprites();
    1 usage
    private static final GhostFactory GHOST_FACTORY = new GhostFactory(PACMAN_SPRITE);

    SwiftNemesis
    @Test
    void doesExist(){
        Ghost inky = GHOST_FACTORY.createInky();
        assertThat(inky).isNotNull();
    }

}
```

Code Coverage:

Coverage | Tests in 'jpacman'

Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	25% (14/55)	13% (43/312)	10% (127/1157)
board	20% (2/10)	9% (5/53)	9% (14/141)
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)	6% (5/78)	3% (13/350)
npc	60% (6/10)	25% (12/47)	12% (31/243)
ghost	55% (5/9)	25% (11/43)	11% (26/235)
Blinky	0% (0/1)	0% (0/4)	0% (0/22)
Clyde	100% (1/1)	50% (2/4)	29% (9/31)
GhostColor	100% (1/1)	100% (1/1)	100% (5/5)
GhostFactory	100% (1/1)	80% (4/5)	85% (6/7)
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	100% (1/1)	50% (2/4)	13% (3/22)
Ghost	100% (1/1)	25% (1/4)	62% (5/8)
points	0% (0/2)	0% (0/7)	0% (0/19)
sprite	66% (4/6)	46% (21/45)	53% (69/128)
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)

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

Github fork: https://github.com/SwiftNemesis/jpacman

Task 3:

Questions:

- Are the coverage results from JaCoCo similar to the ones you got from IntelliJ in the last task? Why so or why not?
    - Answer: The coverage results are not similar at all between the two images. JaCoCo is looking for missed instructions while the intellij report doesn't look for that. You can also see another example where methods in the level package have completely different numbers for how many are actually covered in the testing. JaCoCo treats any branch not covered as not touched while Intellj does which show's why there is a big difference between the two coverage reports.
- Did you find the source code visualization helpful from JaCoCo on uncovered branches?

    - Answer: For JaCoCo, it's confusing as to what the red and green colors mean and it's convoluted what is actually missing. As for the rest, the numbers and coverage percentages are helpful in determining what you need to create unit tests for.
- Which visualization did you prefer and why? IntelliJ's coverage window or JaCoCo's report?
    - Answer: I like the Intellij Coverage window much more. It's not quite as in depth but that's a good thing in terms of being able to digest the information given. That said, the information can then be used to create new unit tests easily and it's also built right into the window.

Task 4:

Github fork: https://github.com/SwiftNemesis/test_coverage

Test Coverage:

```
swift@Swifts-PC:~/test_coverage$ nosetests

Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test deleting an account
- Test finding an account
- Test account from dict
- Test the representation of an account
- Test account to dict
- Test updating an account

Name                 Stmts   Miss  Cover   Missing
--------------------------------------------------
models/__init__.py       7      0   100%
models/account.py       40      0   100%
--------------------------------------------------
TOTAL                   47      0   100%
----------------------------------------------------------------------
Ran 8 tests in 0.412s

OK
```

Code Snippets:

```python
    def test_from_dict(self):
        """ Test account from dict """
        data = ACCOUNT_DATA[self.rand]
        account = Account(**data)
        result = account.from_dict(data)

    def test_update(self):
        """ Test updating an account """
        data = ACCOUNT_DATA[self.rand]
        account = Account(**data)
        account.create()
        account.name = "Foo"
        account.update()
        self.assertEqual(account.name, "Foo")
        with self.assertRaises(DataValidationError):
            account.id = None
            account.update()

    def test_delete(self):
        """ Test deleting an account """
        data = ACCOUNT_DATA[self.rand]
        account = Account(**data)
        account.create()
        account.delete()
        self.assertEqual(len(Account.all()), 0)

    def test_find(self):
        """ Test finding an account """
        data = ACCOUNT_DATA[self.rand]
        account = Account(**data)
        account.create()
        result = Account.find(account.id)
        self.assertEqual(result.id, account.id)
```

Task 5:

Code Snippets:

Counter.py:

```
@app.route('/counters/<name>', methods=['POST'])
def create_counter(name):
    """Create a counter"""
    app.logger.info(f"Request to create counter: {name}")
    global COUNTERS
    if name in COUNTERS:
        return {"Message":f"Counter {name} already exists"}, status.HTTP_409_CONFLICT
    COUNTERS[name] = 0
    return {name: COUNTERS[name]}, status.HTTP_201_CREATED

@app.route('/counters/<name>', methods=['PUT'])
def update_counter(name):
    """Update a counter"""
    app.logger.info(f"Request to update counter: {name}")
    global COUNTERS
    COUNTERS[name] += 1
    return {name: COUNTERS[name]}, status.HTTP_200_OK

@app.route('/counters/<name>', methods=['GET'])
def read_counter(name):
    """Read a counter"""
    app.logger.info(f"Request to read counter: {name}")
    global COUNTERS
    return {name: COUNTERS[name]}, status.HTTP_200_OK
```

Test_counter.py

```python
class CounterTest(TestCase):
    """Counter tests"""
    def setUp(self):
        self.client = app.test_client()

    def test_create_a_counter(self):
        """It should create a counter"""
        client = app.test_client()
        result = client.post('/counters/foo')
        self.assertEqual(result.status_code, status.HTTP_201_CREATED)

    def test_duplicate_a_counter(self):
        """It should return an error for duplicates"""
        result = self.client.post('/counters/bar')
        self.assertEqual(result.status_code, status.HTTP_201_CREATED)
        result = self.client.post('/counters/bar')
        self.assertEqual(result.status_code, status.HTTP_409_CONFLICT)

    def test_update_a_counter(self):
        """It should update a counter"""
        result = self.client.post('/counters/baz')
        self.assertEqual(result.status_code, status.HTTP_201_CREATED)
        result = self.client.put('/counters/baz')
        self.assertEqual(result.status_code, status.HTTP_200_OK)

    def test_read_counter(self):
        """It should read a counter"""
        result = self.client.post('/counters/qux')
        self.assertEqual(result.status_code, status.HTTP_201_CREATED)
        result = self.client.get('/counters/qux')
        self.assertEqual(result.status_code, status.HTTP_200_OK)
```

Testing:

```
swift@Swifts-PC:~/tdd$ nosetests

Counter tests
- It should create a counter

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py        9      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                15      0   100%
----------------------------------------------------------------------
Ran 1 test in 0.073s

OK
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

======================================================================
FAIL: It should return an error for duplicates
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/swift/tdd/tests/test_counter.py", line 37, in test_duplicate_a_counter
    self.assertEqual(result.status_code, status.HTTP_409_CONFLICT)
AssertionError: 201 != 409
-------------------- >> begin captured logging << --------------------
src.counter: INFO: Request to create counter: bar
src.counter: INFO: Request to create counter: bar
-------------------- >> end captured logging << --------------------

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py        9      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                15      0   100%
----------------------------------------------------------------------
Ran 2 tests in 0.068s

FAILED (failures=1)
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       11      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                17      0   100%
----------------------------------------------------------------------
Ran 2 tests in 0.070s

OK
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

======================================================================
FAIL: It should update a counter
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/swift/tdd/tests/test_counter.py", line 44, in test_update_a_counter
    self.assertEqual(result.status_code, status.HTTP_200_OK)
AssertionError: 405 != 200
-------------------- >> begin captured logging << --------------------
src.counter: INFO: Request to create counter: baz
--------------------- >> end captured logging << ---------------------

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       11      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                17      0   100%
----------------------------------------------------------------------
Ran 3 tests in 0.079s

FAILED (failures=1)
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       18      1    94%   30
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                24      1    96%
----------------------------------------------------------------------
Ran 3 tests in 0.089s

OK
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

======================================================================
FAIL: It should read a counter
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/swift/tdd/tests/test_counter.py", line 51, in test_read_counter
    self.assertEqual(result.status_code, status.HTTP_200_OK)
AssertionError: 405 != 200
-------------------- >> begin captured logging << --------------------
src.counter: INFO: Request to create counter: qux
--------------------- >> end captured logging << ---------------------

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       16      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                22      0   100%
----------------------------------------------------------------------
Ran 4 tests in 0.087s

FAILED (failures=1)
```

```
swift@Swifts-PC:~/tdd$ nosetests

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

Name              Stmts   Miss  Cover   Missing
-----------------------------------------------
src/counter.py       20      0   100%
src/status.py         6      0   100%
-----------------------------------------------
TOTAL                26      0   100%
----------------------------------------------------------------------
Ran 4 tests in 0.075s

OK
```

Github Fork: https://github.com/SwiftNemesis/tdd