Antonio Tessier
CS 472-1001
February 6, 2024

Testing Lab

Task 2.1

Below are tests I designed for JPacman with path references. Also included are coverage percentages before and after they were added.

src/test/java/nl/tudelft/jpacman/level/CollisionInteractionMapTest.java

```
@Test
void testCollisionInteractionMap()
{
    /**
     *  Antonio Tessier
     *
     *  Create a collisionMap
     *  Check if it's null by default
     *  Give the map default collisions
     *  Check if the map is no longer null
     */

    CollisionInteractionMap collisionMap = null;
    assertThat(collisionMap).isNull();
    collisionMap = defaultCollisions();
    assertThat(collisionMap).isNotNull();

}
```

Class%(12 ->16), Method%(8 ->10), Line%(7->9)

src/test/java/nl/tudelft/jpacman/level/createPelletTest.java

```
@Test
void testCreatePellet()
{
    /**
     *  Antonio Tessier
     *
     *  Build a Level Factory
     *  Create a pellet
     *  Check if the pellet is null
     *  Have Level Factory assign the pellet
     *  Check if the pellet is no longer null
     */
    PacManSprites sprites = new PacManSprites();    // nl/tudelft/jpacman/npc/ghost/NavigationTest.java
    LevelFactory levelFactory = new LevelFactory(   // Line 42
        sprites,
        new GhostFactory(sprites),
        mock(PointCalculator.class));

    Pellet pellet = null;
    assertThat(pellet).isNull();
    pellet = levelFactory.createPellet();
    assertThat(pellet).isNotNull();

}
```

Class%(16->22), Method%(10->11), Line%(9->10)

src/test/java/nl/tudelft/jpacman/level/levelStartTest.java

```java
@Test
void testLevelStart()
{
    /**
     *  Antonio Tessier
     *
     *  Start the level
     *  Check if it's in progress
     *  Stop the level
     *  Check if it's not in progress
     */
    level.start();
    assertThat(level.isInProgress()).isEqualTo( expected: true);
    level.stop();
    assertThat(level.isInProgress()).isEqualTo( expected: false);

}
```

Class%(22->29), Method%(11->16), Line%(10->15)

src/test/java/nl/tudelft/jpacman/level/levelStopTest.java

```java
@Test
void testLevelStop()
{
    /**
     *  Antonio Tessier
     *
     *  Stop the level
     *  Check if it's not in progress
     *  Start the level
     *  Check if it's in progress
     */
    level.stop();
    assertThat(level.isInProgress()).isEqualTo( expected: false);
    level.start();
    assertThat(level.isInProgress()).isEqualTo( expected: true);

}
```

Class%(29->29), Method%(16->16), Line%(15->16)

Task 3

IntelliJ's built-in Coverage report

Current scope: all classes

Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	43.6% (24/55)	23.6% (77/326)	20.2% (242/1197)

Coverage Breakdown

Package	Class, %	Method, %	Line, %
nl.tudelft.jpacman	0% (0/3)	0% (0/29)	0% (0/77)
nl.tudelft.jpacman.board	40% (4/10)	12.5% (7/56)	10.7% (16/149)
nl.tudelft.jpacman.fuzzer	0% (0/1)	0% (0/7)	0% (0/33)
nl.tudelft.jpacman.game	0% (0/3)	0% (0/14)	0% (0/37)
nl.tudelft.jpacman.integration	0% (0/1)	0% (0/5)	0% (0/7)
nl.tudelft.jpacman.level	61.5% (8/13)	38% (30/79)	31% (115/371)
nl.tudelft.jpacman.npc	100% (1/1)	25% (1/4)	62.5% (5/8)
nl.tudelft.jpacman.npc.ghost	66.7% (6/9)	31.8% (14/44)	12.7% (30/236)
nl.tudelft.jpacman.points	0% (0/2)	0% (0/9)	0% (0/21)
nl.tudelft.jpacman.sprite	83.3% (5/6)	52.1% (25/48)	58% (76/131)
nl.tudelft.jpacman.ui	0% (0/6)	0% (0/31)	0% (0/127)

generated on 2024-02-06 15:26

JaCoCo's Coverage report

jpacman Sessions

jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
nl.tudelft.jpacman.level		72%		58%	68	155	86	344	15	69	2	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	1,148 of 4,694	75%	291 of 637	54%	287	590	211	1,039	45	268	4	47

Created with JaCoCo 0.8.3.201901230119

- JaCoCo and IntelliJ have similar results, but appear to vary due to different measuring criteria for covered lines.
- JaCoco's branch coverage visuals are helpful considering IntelliJ does not show branch coverage unless you open the class files and look manually.
- Personally I prefer JaCoCo due to the branch coverage information and the red/green bar showing coverage.

Task 4

Below are code snippets of tests for account.py from test_account.py alongside test coverage outputs.

Initial test coverage (68%)

```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data

Name                 Stmts   Miss  Cover   Missing
--------------------------------------------------
models\__init__.py       7      0   100%
models\account.py       40     13    68%   26, 30, 34-35, 45-48, 52-54, 74-75
--------------------------------------------------
TOTAL                   47     13    72%
----------------------------------------------------------------------
Ran 2 tests in 0.437s
```

```python
def test_repr(self):
    """Test the representation of an account"""
    account = Account()
    account.name = "Foo"
    self.assertEqual(str(account), "<Account 'Foo'>")
```

Test coverage after adding test_repr (74%)

```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test the representation of an account

Name                 Stmts   Miss  Cover   Missing
--------------------------------------------------
models\__init__.py       7      0   100%
models\account.py       40     12    70%   30, 34-35, 45-48, 52-54, 74-75
--------------------------------------------------
TOTAL                   47     12    74%
----------------------------------------------------------------------
Ran 3 tests in 0.467s
```

```python
def test_to_dict(self):
    """Test account to dict"""
    data = ACCOUNT_DATA[self.rand] # Get rand account
    account = Account(**data)
    result = account.to_dict()
    self.assertEqual(account.name, result["name"])
    self.assertEqual(account.email, result["email"])
    self.assertEqual(account.phone_number, result["phone_number"])
    self.assertEqual(account.disabled, result["disabled"])
    self.assertEqual(account.date_joined, result["date_joined"])
```

Test coverage after adding test_to_dict (77%)

```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test the representation of an account
- Test account to dict

Name                 Stmts   Miss  Cover   Missing
--------------------------------------------------
models\__init__.py       7      0   100%
models\account.py       40     11    72%   34-35, 45-48, 52-54, 74-75
--------------------------------------------------
TOTAL                   47     11    77%
----------------------------------------------------------------------
Ran 4 tests in 0.448s
```

```python
def test_from_dict(self):
    """Test account from dict"""

    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)
    result = {"name": "", "email": "", "phone_number": "", "disabled": "", "date_joined": ""}
    account.from_dict(result)

    self.assertEqual(result["name"], account.name)
    self.assertEqual(result["email"], account.email)
    self.assertEqual(result["phone_number"], account.phone_number)
    self.assertEqual(result["disabled"], account.disabled)
    self.assertEqual(result["date_joined"], account.date_joined)
```

Test coverage after adding test_from_dict (81%)

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

Name                 Stmts   Miss  Cover   Missing
--------------------------------------------------
models\__init__.py       7      0   100%
models\account.py       40      9    78%   45-48, 52-54, 74-75
--------------------------------------------------
TOTAL                   47      9    81%
----------------------------------------------------------------------
Ran 5 tests in 0.471s
```

```python
def test_update(self):
    """Test account update"""

    data = ACCOUNT_DATA[self.rand]
    account = Account(**data)
    result = account.to_dict()

    account.name = "Antonio Tessier"
    account.email = "tessier@unlv.nevada.edu"
    account.phone_number = "12345"
    account.disabled = "false"
    account.date_joined = "1/1/2001"
    account.id = ""

    with self.assertRaises(DataValidationError):
        account.update()

    account.id = "1"
    account.update()

    self.assertEqual("Antonio Tessier", account.name)
    self.assertEqual("tessier@unlv.nevada.edu", account.email)
    self.assertEqual("12345", account.phone_number)
    self.assertEqual("false", account.disabled)
    self.assertEqual("1/1/2001", account.date_joined)
    self.assertEqual("1", account.id)
```

Test coverage after adding test_update (89%)

```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test account from dict
- Test the representation of an account
- Test account to dict
- Test account update

Name                Stmts   Miss  Cover   Missing
-------------------------------------------------
models\__init__.py      7      0   100%
models\account.py      40      5    88%   52-54, 74-75
-------------------------------------------------
TOTAL                  47      5    89%
----------------------------------------------------------------------
Ran 6 tests in 0.479s
```

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

    account.id = "1"
    account.create()
    self.assertIsNotNone(account)
    account.delete()
    self.assertIsNone(Account.find("1"))
```

Test coverage after adding test_delete (100%)

```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test account deletion
- Test account from dict
- Test the representation of an account
- Test account to dict
- Test account update

Name                Stmts   Miss  Cover   Missing
-------------------------------------------------
models\__init__.py      7      0   100%
models\account.py      40      0   100%
-------------------------------------------------
TOTAL                  47      0   100%
----------------------------------------------------------------------
Ran 7 tests in 0.455s
```

Task 5

Test_create_a_counter inserted into test_counter.py.

```python
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)
```

Now the program is in the **RED** phase.

```
Counter tests
- It should create a counter (FAILED)

======================================================================
FAIL: It should create a counter
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\thegr\IdeaProjects\PythonTesting\tdd\tests\test_counter.py", line 30, in test_create_a_counter
    self.assertEqual(result.status_code, status.HTTP_201_CREATED)
AssertionError: 404 != 201

Name             Stmts   Miss  Cover   Missing
----------------------------------------------
src\counter.py       3      0   100%
src\status.py        6      0   100%
----------------------------------------------
TOTAL                9      0   100%
----------------------------------------------------------------------
Ran 1 test in 0.168s

FAILED (failures=1)
```

create_counter inserted into counter.py.

```python
@app.route('/counters/<name>', methods=['POST'])
def create_counter(name):
    """Create a counter"""
    app.logger.info(f"Request to create counter: {name}")
    global COUNTERS

    COUNTERS[name] = 0
    return {name: COUNTERS[name]}, status.HTTP_201_CREATED
```

Now the program is in a **GREEN** phase. Counter.py is now able to be **REFACTORED**.

```
Counter tests
- It should create a counter

Name             Stmts   Miss  Cover   Missing
------------------------------------------------
src\counter.py      11      1    91%   20
src\status.py        6      0   100%
------------------------------------------------
TOTAL               17      1    94%
----------------------------------------------------------------------
Ran 1 test in 0.179s

OK
```

test_duplicate_a_counter is added into test_counter.py.

```
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)
```

Program is again in a **RED** phase and can no longer be **REFACTORED**.

```
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 "C:\Users\thegr\IdeaProjects\PythonTesting\tdd\tests\test_counter.py", line 41, 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      19      6    68%   27-31, 36-40
src\status.py        6      0   100%
------------------------------------------------
TOTAL               25      6    76%
----------------------------------------------------------------------
Ran 2 tests in 0.171s

FAILED (failures=1)
```

create_counter is modified to handle duplicate names.

```python
@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
```

counter.py is back in a **GREEN** phase and can be **REFACTORED** again.

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

Name             Stmts   Miss  Cover   Missing
------------------------------------------------
src\counter.py      21      6    71%   28-32, 37-41
src\status.py        6      0   100%
------------------------------------------------
TOTAL               27      6    78%
----------------------------------------------------------------------
Ran 2 tests in 0.172s

OK
```

test_update_a_counter is added to test_counter.py.

```python
def test_update_a_counter(self):
    """It should update a counter"""
    client = app.test_client()
    result = client.post('/counters/foo2')
    self.assertEqual(result.status_code, status.HTTP_201_CREATED)

    result = self.client.put('/counters/foo2')

    self.assertEqual(result.status_code, status.HTTP_200_OK)
    self.assertEqual(result.json['foo2'], 1)
```

Back to **RED** phase.

```
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 "C:\Users\thegr\IdeaProjects\PythonTesting\tdd\tests\test_counter.py", line 51, 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: foo2
--------------------- >> 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 0.174s

FAILED (failures=1)
```

update_counter added to counter.py.

```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
    COUNTERS[name] += 1

    return {name: COUNTERS[name]}, status.HTTP_200_OK
```

Back to **GREEN** phase, can **REFACTOR** counter.py again.

```
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      21      3    86%   37-41
src\status.py        6      0   100%
-----------------------------------------------
TOTAL               27      3    89%
----------------------------------------------------------------------
Ran 3 tests in 0.175s

OK
```

test_read_a_counter added to test_counter.py.

```python
def test_read_a_counter(self):
    """It should read a counter"""
    client = app.test_client()
    result = client.post('/counters/foo3')
    self.assertEqual(result.status_code, status.HTTP_201_CREATED)

    result = self.client.get('/counters/foo3')

    self.assertEqual(result.status_code, status.HTTP_200_OK)
    self.assertEqual(result.json['foo3'], 0)
```

Back to **RED** phase.

```
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 "C:\Users\thegr\IdeaProjects\PythonTesting\tdd\tests\test_counter.py", line 62, in test_read_a_counter
    self.assertEqual(result.status_code, status.HTTP_200_OK)
AssertionError: 405 != 200
-------------------- >> begin captured logging << --------------------
src.counter: INFO: Request to create counter: foo3
--------------------- >> 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 0.164s

FAILED (failures=1)
```

read_counter added to counter.py.

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

Back to **GREEN** phase, can **REFACTOR** counter.py again.

```
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      21      0   100%
src\status.py        6      0   100%
------------------------------------------------
TOTAL               27      0   100%
----------------------------------------------------------------------
Ran 4 tests in 0.163s

OK
```

Errors:

- AssertionError: 404 != 201
  - No Endpoint called “/counters”, added “/counters”
- AssertionError: 201 != 409
  - Name conflict. Added condition for checking duplicate counter names