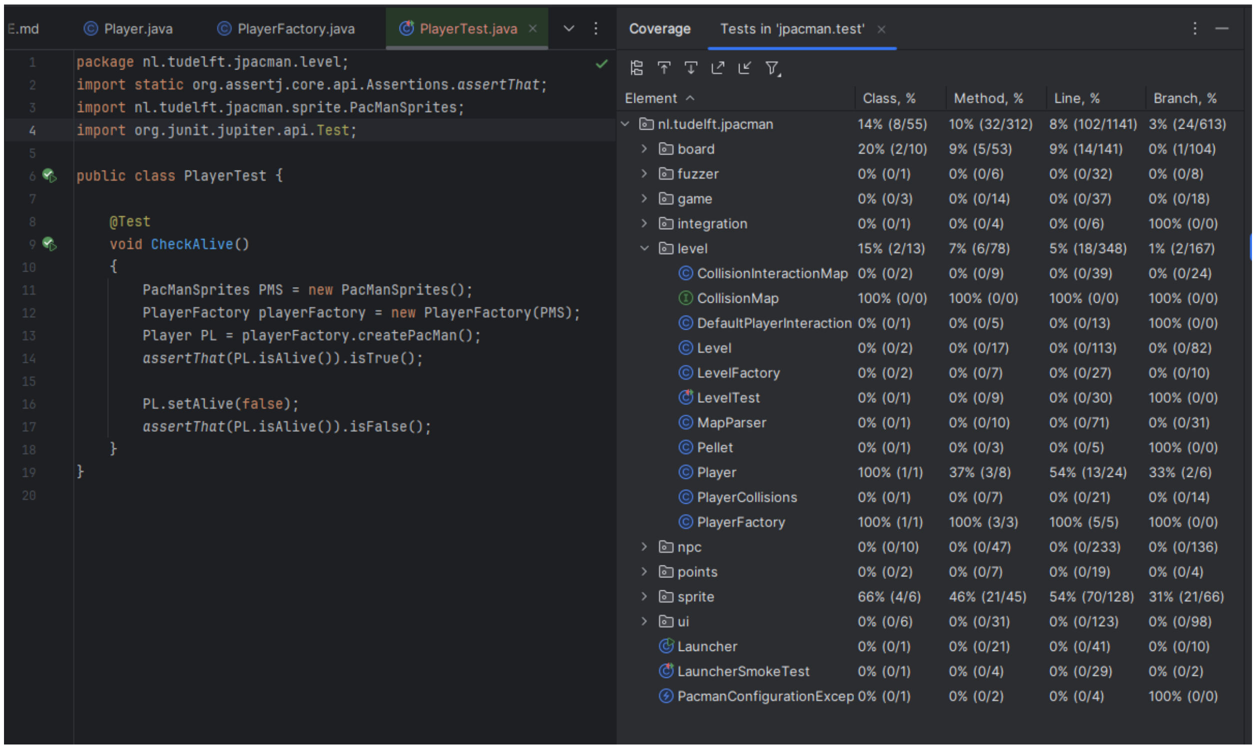
Aarush Misherwan
CS 472
9/17/2024
link: https://github.com/Group3-472/Main/tree/main/jpacman

Task 2

By writing a test case to test the method isAlive() from player, there is a noticeable increase to the test coverage results. As you can see below from *Figure 1,* the increase comes from adding test to the Player class. The PlayerTest.java file shows the test case the was written about. If it returns true, then the player is alive. If false, then the player is dead.



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

public class PlayerTest {

    @Test
    void CheckAlive()
    {
        PacManSprites PMS = new PacManSprites();
        PlayerFactory playerFactory = new PlayerFactory(PMS);
        Player PL = playerFactory.createPacMan();
        assertThat(PL.isAlive()).isTrue();

        PL.setAlive(false);
        assertThat(PL.isAlive()).isFalse();
    }
}
```

Element	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.jpacman	14% (8/55)	10% (32/312)	8% (102/1141)	3% (24/613)
board	20% (2/10)	9% (5/53)	9% (14/141)	0% (1/104)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	15% (2/13)	7% (6/78)	5% (18/348)	1% (2/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteraction	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)	0% (0/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)
npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/136)
points	0% (0/2)	0% (0/7)	0% (0/19)	0% (0/4)
sprite	66% (4/6)	46% (21/45)	54% (70/128)	31% (21/66)
ui	0% (0/6)	0% (0/31)	0% (0/123)	0% (0/98)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/10)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
PacmanConfigurationExcep	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)

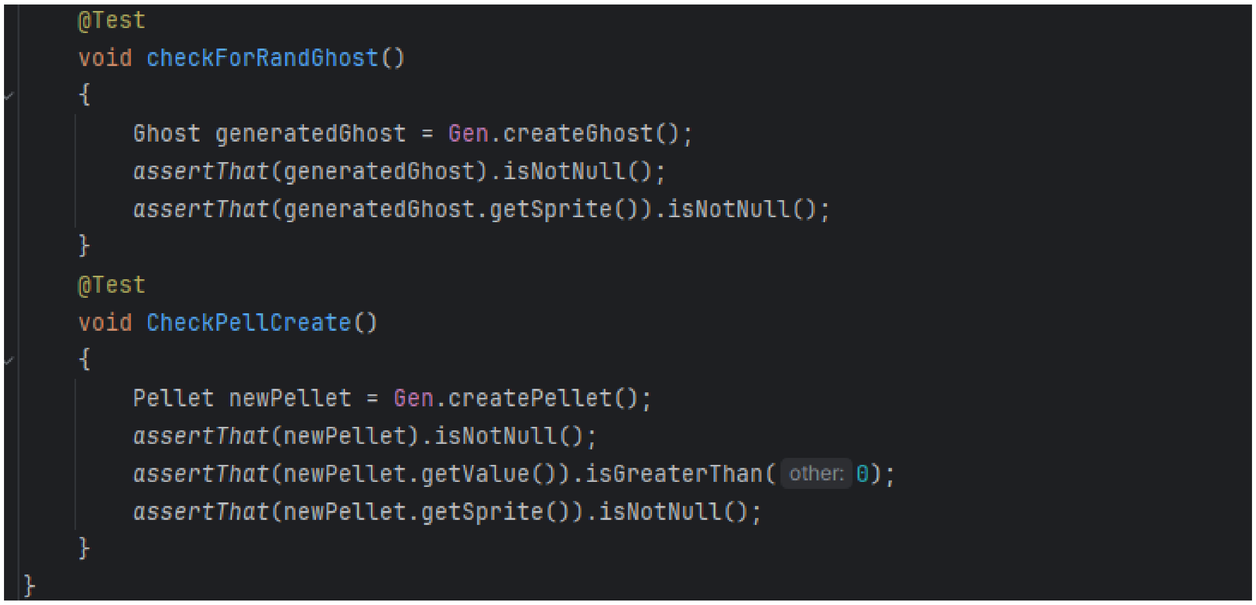
Figure1: The test case for testing the method isAlive()

Task 2.1

I did four methods in total. The first two methods were pellet and getValue from the Pellet file. The last two methods were createPellet and RandomGhost, both from the LevelFactory file. *Figure 3* and *figure 6* will show the code snippets for their methods. Below the code snippets are the test coverage results on the other figures.

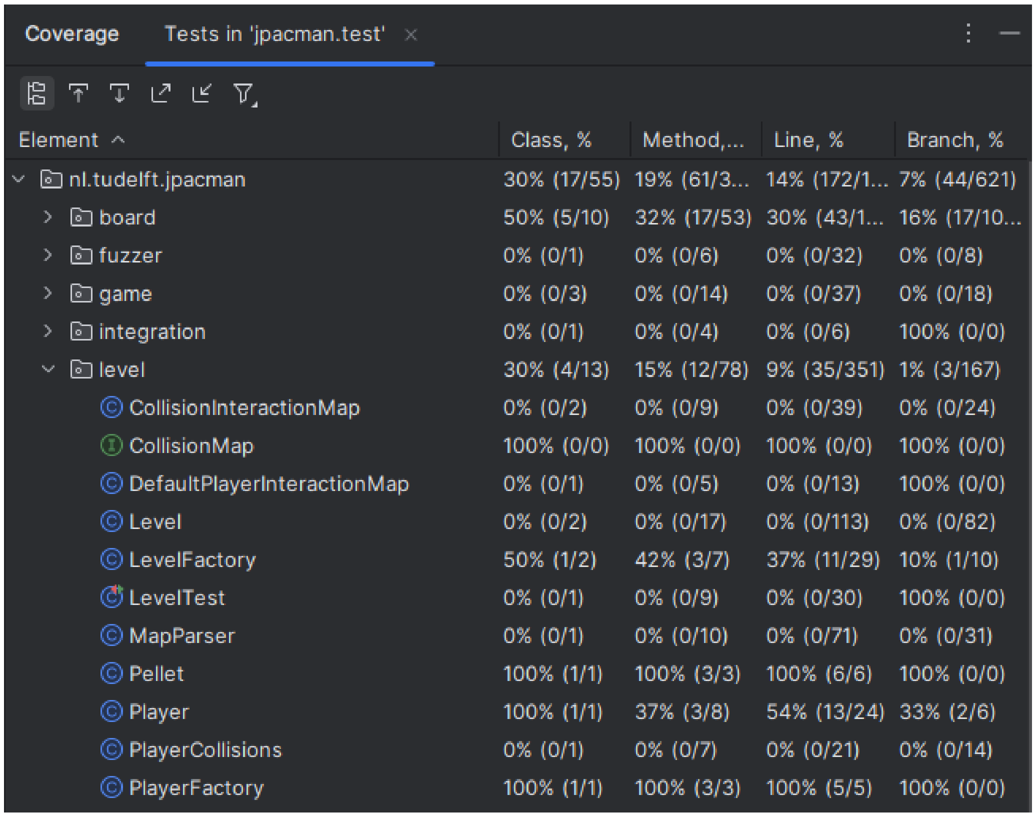
Task 2.1: Methods createPellet and RandomGhost

The orginal coverage results can be seen on *figure 3* for the LevelFactory class. The after results by adding a test case createPellet can be seen in *figure 4*. Then by adding a test case for RandomGhost, the coverage results are seen in *figure 5*.



```
    @Test
    void checkForRandGhost()
    {
        Ghost generatedGhost = Gen.createGhost();
        assertThat(generatedGhost).isNotNull();
        assertThat(generatedGhost.getSprite()).isNotNull();
    }
    @Test
    void CheckPellCreate()
    {
        Pellet newPellet = Gen.createPellet();
        assertThat(newPellet).isNotNull();
        assertThat(newPellet.getValue()).isGreaterThan( other: 0);
        assertThat(newPellet.getSprite()).isNotNull();
    }
}
```

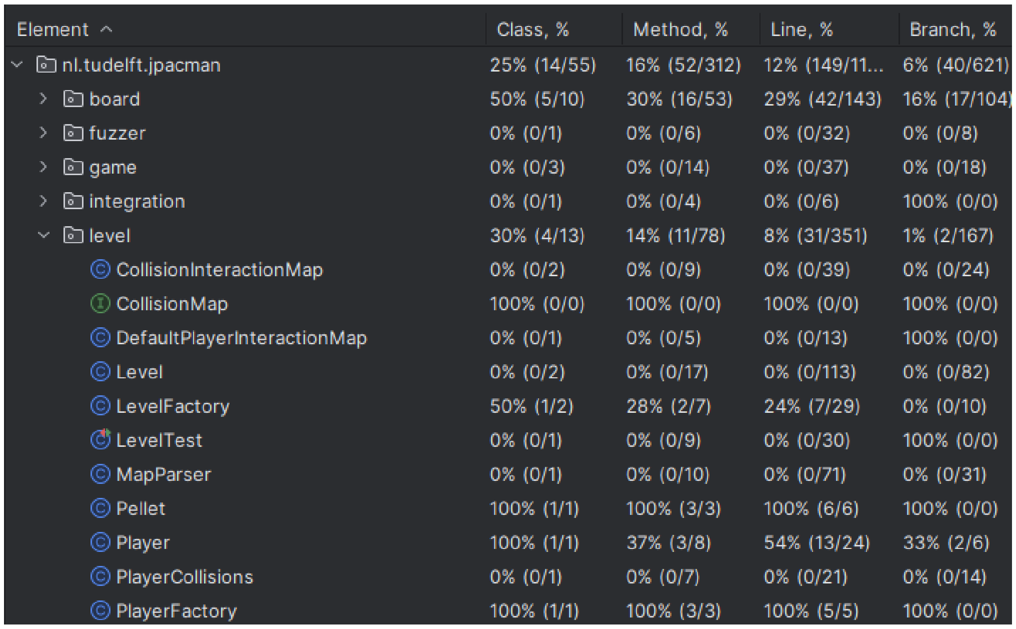
Figure 2: Test cases for the methods createPellet and RandomGhost



Coverage   Tests in 'jpacman.test'

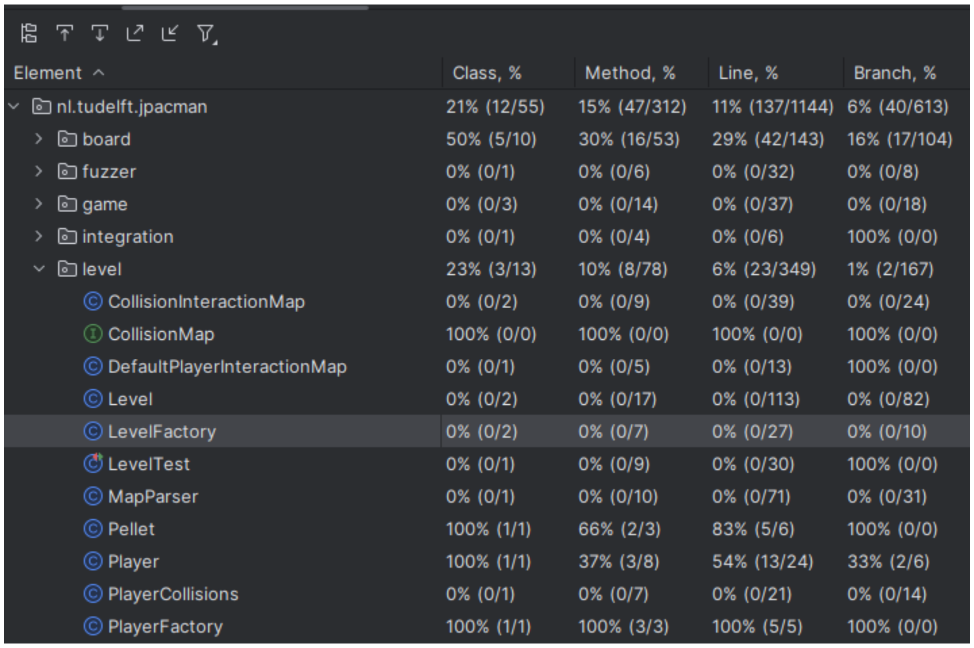
Element	Class, %	Method,...	Line, %	Branch, %
nl.tudelft.jpacman	30% (17/55)	19% (61/3...	14% (172/1...	7% (44/621)
board	50% (5/10)	32% (17/53)	30% (43/1...	16% (17/10...
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	30% (4/13)	15% (12/78)	9% (35/351)	1% (3/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	50% (1/2)	42% (3/7)	37% (11/29)	10% (1/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	100% (1/1)	100% (3/3)	100% (6/6)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)

Figure 3: LevelFactor file original



Element	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.jpacman	25% (14/55)	16% (52/312)	12% (149/11...	6% (40/621)
board	50% (5/10)	30% (16/53)	29% (42/143)	16% (17/104)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	30% (4/13)	14% (11/78)	8% (31/351)	1% (2/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	50% (1/2)	28% (2/7)	24% (7/29)	0% (0/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	100% (1/1)	100% (3/3)	100% (6/6)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)

Figure 4: LevelFactory after making test case for createPellet method

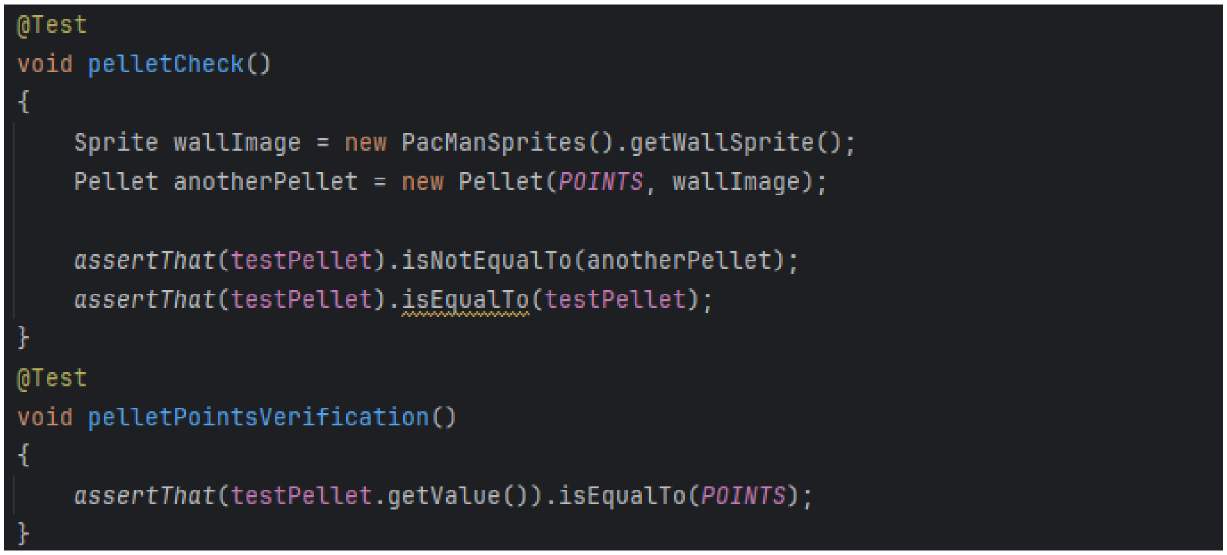


Element	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.jpacman	21% (12/55)	15% (47/312)	11% (137/1144)	6% (40/613)
board	50% (5/10)	30% (16/53)	29% (42/143)	16% (17/104)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	23% (3/13)	10% (8/78)	6% (23/349)	1% (2/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)	0% (0/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	100% (1/1)	66% (2/3)	83% (5/6)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)

Figure 5: LevelFactory after making test case for RandomGhost method

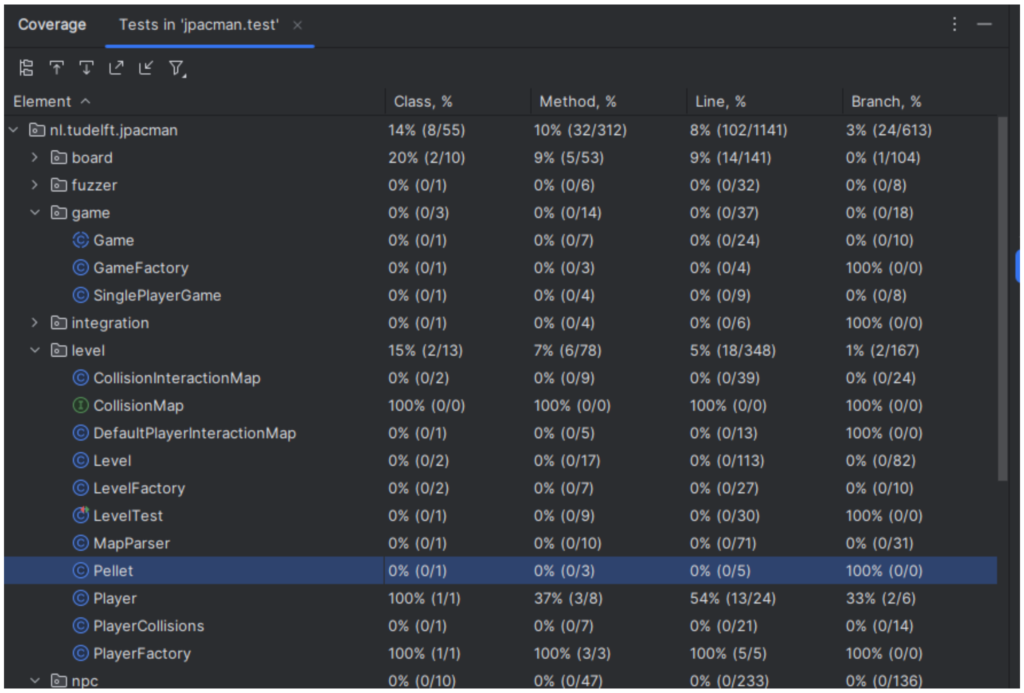
Task 2.1: Methods pellet and getValue

The code snippet can be seen below in *Figure* 6. The results by adding test cases for pellet and getValue can be seen below in *figure 8*. To see the original coverage results for the Pellet class, it can be seen below in *figure 7*.



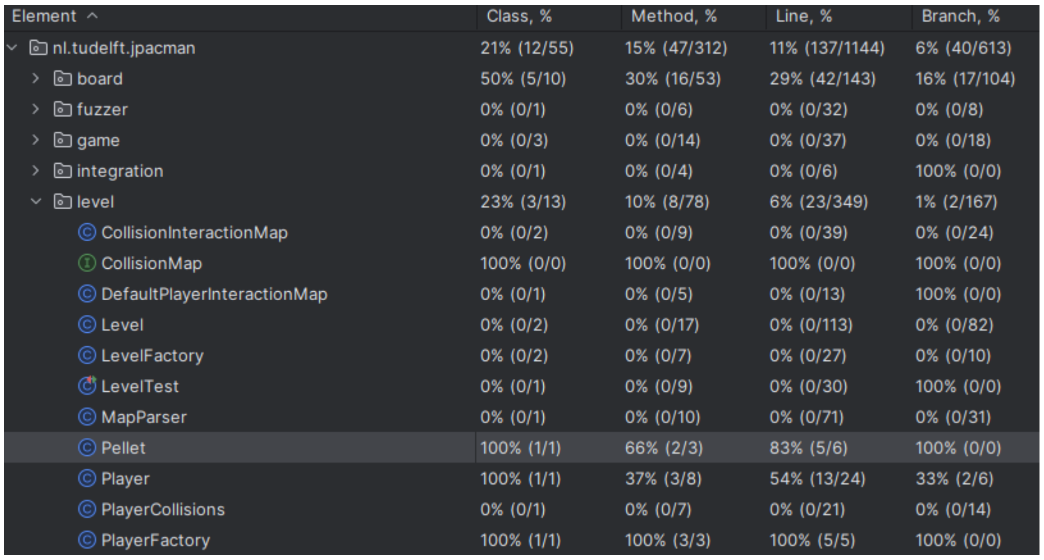
```java
@Test
void pelletCheck()
{
    Sprite wallImage = new PacManSprites().getWallSprite();
    Pellet anotherPellet = new Pellet(POINTS, wallImage);

    assertThat(testPellet).isNotEqualTo(anotherPellet);
    assertThat(testPellet).isEqualTo(testPellet);
}
@Test
void pelletPointsVerification()
{
    assertThat(testPellet.getValue()).isEqualTo(POINTS);
}
```

Figure 6: Code snippet for test case for methods pellet and getValue.



Element	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.jpacman	14% (8/55)	10% (32/312)	8% (102/1141)	3% (24/613)
board	20% (2/10)	9% (5/53)	9% (14/141)	0% (1/104)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
Game	0% (0/1)	0% (0/7)	0% (0/24)	0% (0/10)
GameFactory	0% (0/1)	0% (0/3)	0% (0/4)	100% (0/0)
SinglePlayerGame	0% (0/1)	0% (0/4)	0% (0/9)	0% (0/8)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	15% (2/13)	7% (6/78)	5% (18/348)	1% (2/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)	0% (0/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	0% (0/1)	0% (0/3)	0% (0/5)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)
npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/136)

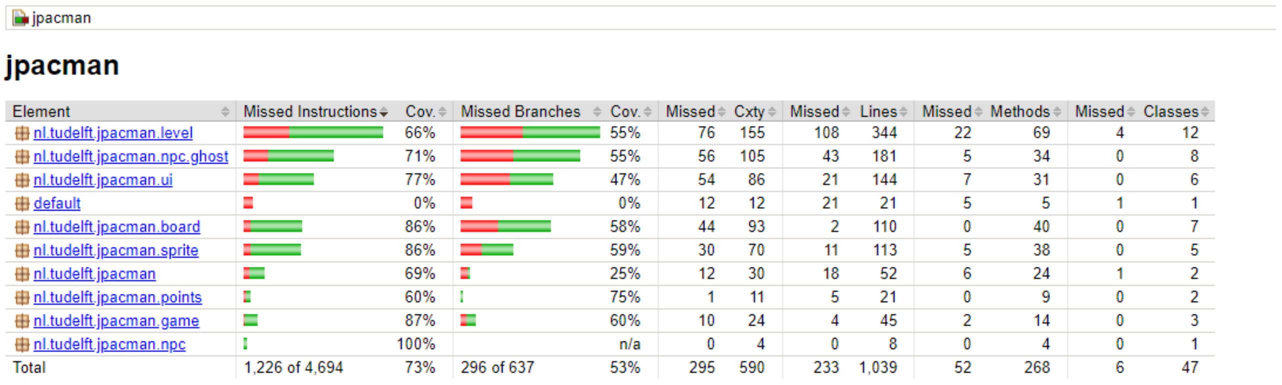
Figure 7: Pellet file original



Element	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.jpacman	21% (12/55)	15% (47/312)	11% (137/1144)	6% (40/613)
board	50% (5/10)	30% (16/53)	29% (42/143)	16% (17/104)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/18)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	23% (3/13)	10% (8/78)	6% (23/349)	1% (2/167)
CollisionInteractionMap	0% (0/2)	0% (0/9)	0% (0/39)	0% (0/24)
CollisionMap	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
DefaultPlayerInteractionMap	0% (0/1)	0% (0/5)	0% (0/13)	100% (0/0)
Level	0% (0/2)	0% (0/17)	0% (0/113)	0% (0/82)
LevelFactory	0% (0/2)	0% (0/7)	0% (0/27)	0% (0/10)
LevelTest	0% (0/1)	0% (0/9)	0% (0/30)	100% (0/0)
MapParser	0% (0/1)	0% (0/10)	0% (0/71)	0% (0/31)
Pellet	100% (1/1)	66% (2/3)	83% (5/6)	100% (0/0)
Player	100% (1/1)	37% (3/8)	54% (13/24)	33% (2/6)
PlayerCollisions	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/14)
PlayerFactory	100% (1/1)	100% (3/3)	100% (5/5)	100% (0/0)

Figure 8: Pellet file after adding test cases for pellet and getValue

Task 3


jpacman

jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
nl.tudelft.jpacman.level		66%		55%	76	155	108	344	22	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	1,226 of 4,694	73%	296 of 637	53%	295	590	233	1,039	52	268	6	47

Figure 9: JaCoCo Report on JPacman

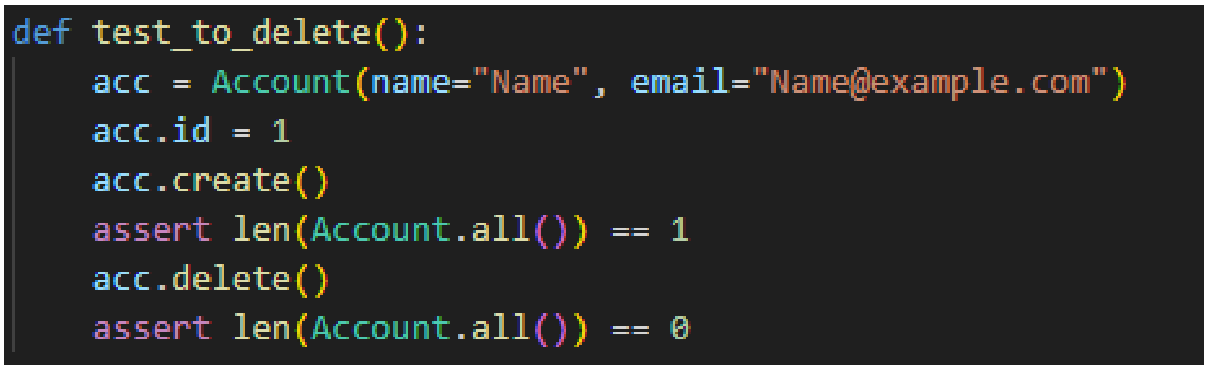
1. As you can see from the JaCoCo report above in *Figure 9,* the coverage results are not similar. It's because JaCoCo reports on what is missed percentage, such as for the Instructions, branches and classes. While IntelliJ reports on what on the coverage's percentage for the same instructions, branches and classes.

2. I do find the JaCoCo report helpful as it helps identify branches that could be tested otherwise it is considered missing.

3. I prefer IntelliJ's because it is connected to IntelliJ's IDE and opens a window to show the coverage data. I can also double click on any of the java classes and it opens a window to them. This makes it easier for me to debug and have what I need in one application. But I do like JaCoCo's for how it shows what lines or branches are not covered.

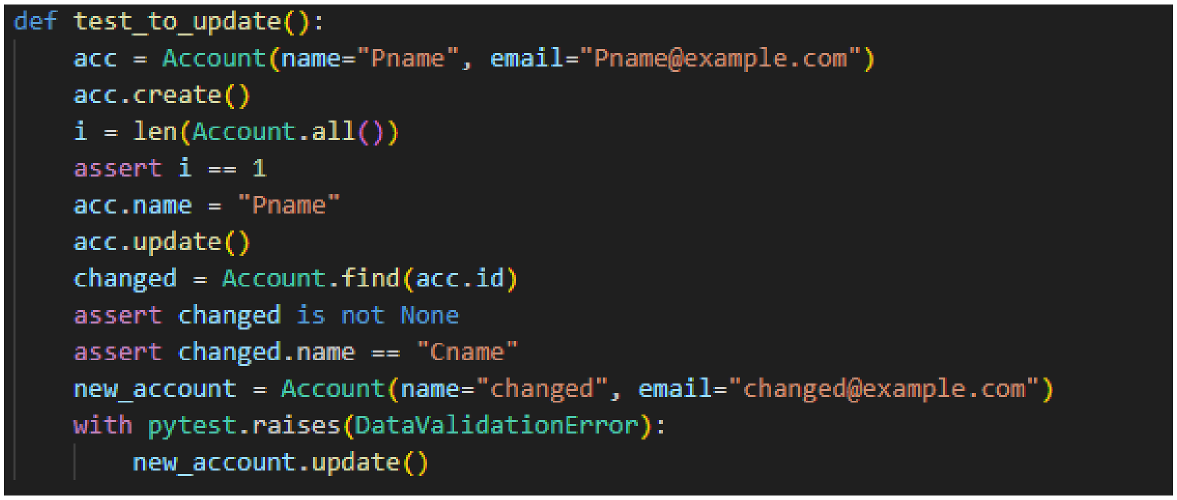
Task 4

First, I worked on testing the delete method, which will create a name and email address for an account, then delete it. By adding the account to the database, it can check if the test database has an account, which should be 1. Then we delete it to make sure it was deleted, resulting in no accounts in the database, which can be seen in *Figure 10*. Next was to test the update method, done by creating an account with a name and email, placing it into the database and calling an update method to change the name and email, which is the update. So now we check if the database has the name and email changed. This can be seen in *Figure 11*. Last was to check the from_dict method, which is done by creating a dict and checking the values, which is seen in *Figure 12*. The coverage results are seen in *Figure 13*.



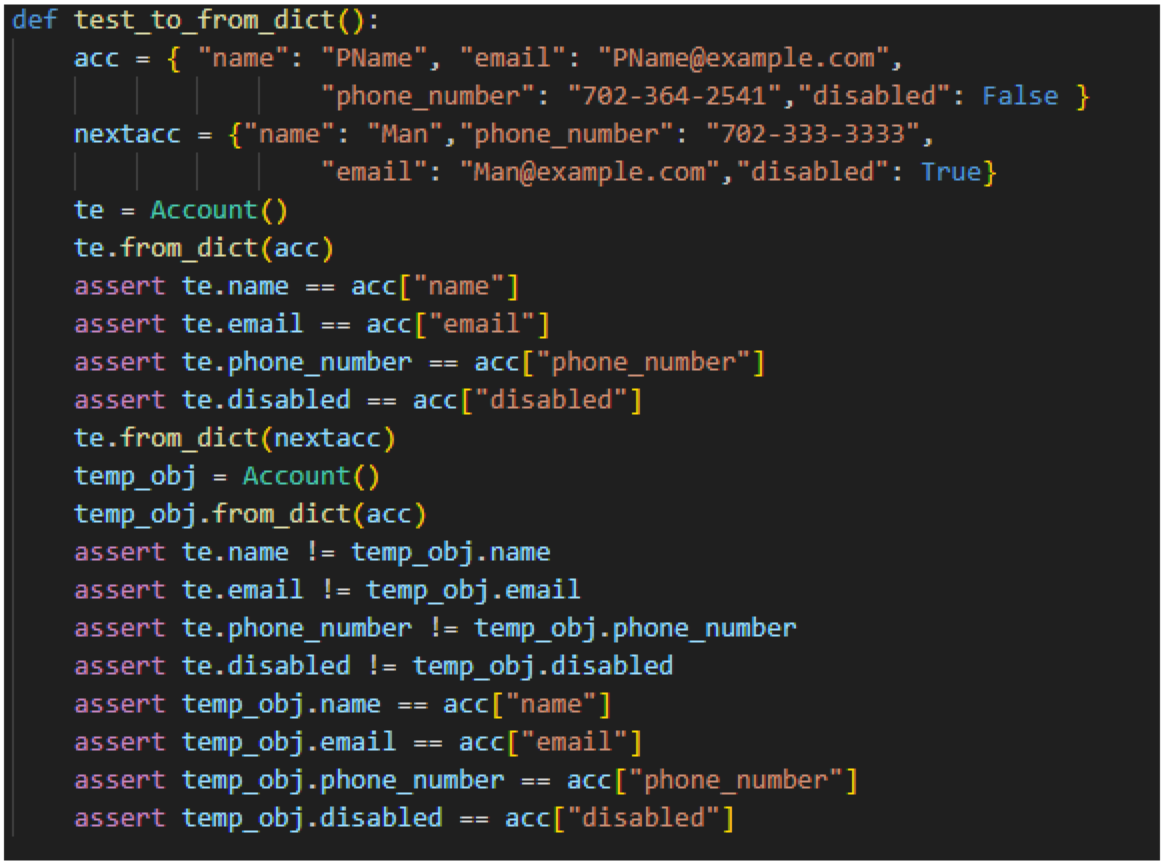
```python
def test_to_delete():
    acc = Account(name="Name", email="Name@example.com")
    acc.id = 1
    acc.create()
    assert len(Account.all()) == 1
    acc.delete()
    assert len(Account.all()) == 0
```

Figure 10: Test Delete



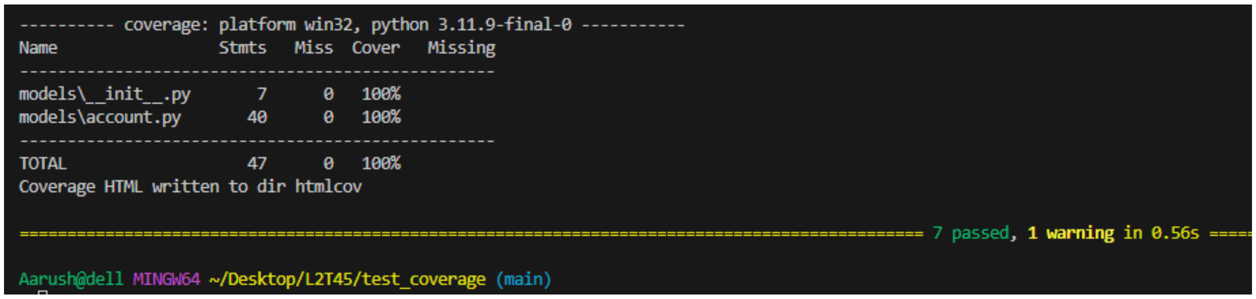
```python
def test_to_update():
    acc = Account(name="Pname", email="Pname@example.com")
    acc.create()
    i = len(Account.all())
    assert i == 1
    acc.name = "Pname"
    acc.update()
    changed = Account.find(acc.id)
    assert changed is not None
    assert changed.name == "Cname"
    new_account = Account(name="changed", email="changed@example.com")
    with pytest.raises(DataValidationError):
        new_account.update()
```

Figure 11: Test update



```python
def test_to_from_dict():
    acc = { "name": "PName", "email": "PName@example.com",
                "phone_number": "702-364-2541","disabled": False }
    nextacc = {"name": "Man","phone_number": "702-333-3333",
                "email": "Man@example.com","disabled": True}
    te = Account()
    te.from_dict(acc)
    assert te.name == acc["name"]
    assert te.email == acc["email"]
    assert te.phone_number == acc["phone_number"]
    assert te.disabled == acc["disabled"]
    te.from_dict(nextacc)
    temp_obj = Account()
    temp_obj.from_dict(acc)
    assert te.name != temp_obj.name
    assert te.email != temp_obj.email
    assert te.phone_number != temp_obj.phone_number
    assert te.disabled != temp_obj.disabled
    assert temp_obj.name == acc["name"]
    assert temp_obj.email == acc["email"]
    assert temp_obj.phone_number == acc["phone_number"]
    assert temp_obj.disabled == acc["disabled"]
```

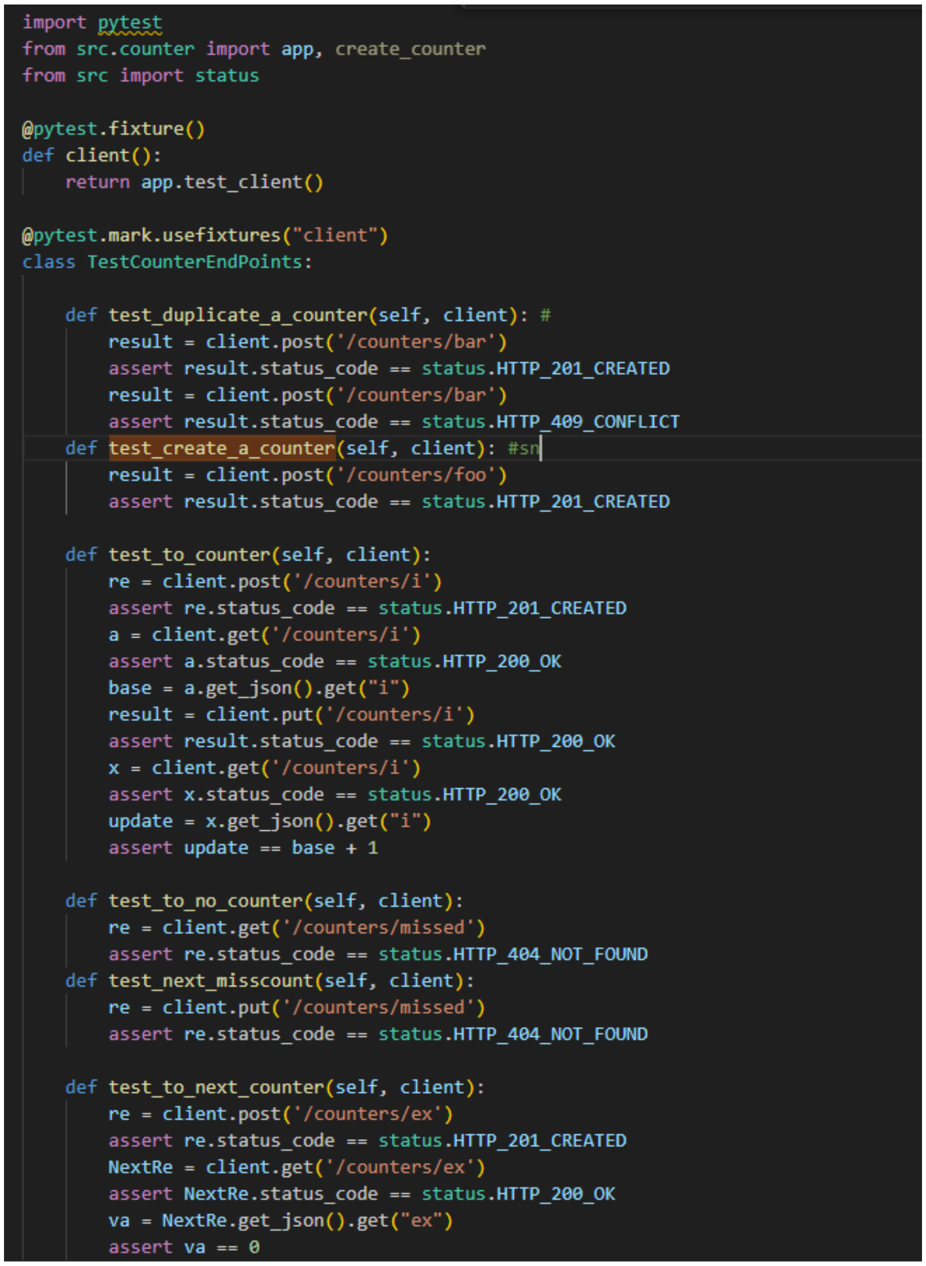
Figure 12: Test From Dict



```
---------- coverage: platform win32, python 3.11.9-final-0 -----------
Name                   Stmts   Miss  Cover   Missing
----------------------------------------------------
models\__init__.py         7      0   100%
models\account.py         40      0   100%
----------------------------------------------------
TOTAL                     47      0   100%
Coverage HTML written to dir htmlcov

=========================================================================================== 7 passed, 1 warning in 0.56s ====

Aarush@dell MINGW64 ~/Desktop/L2T45/test_coverage (main)
```

Figure 13: Coverage results

Task 5



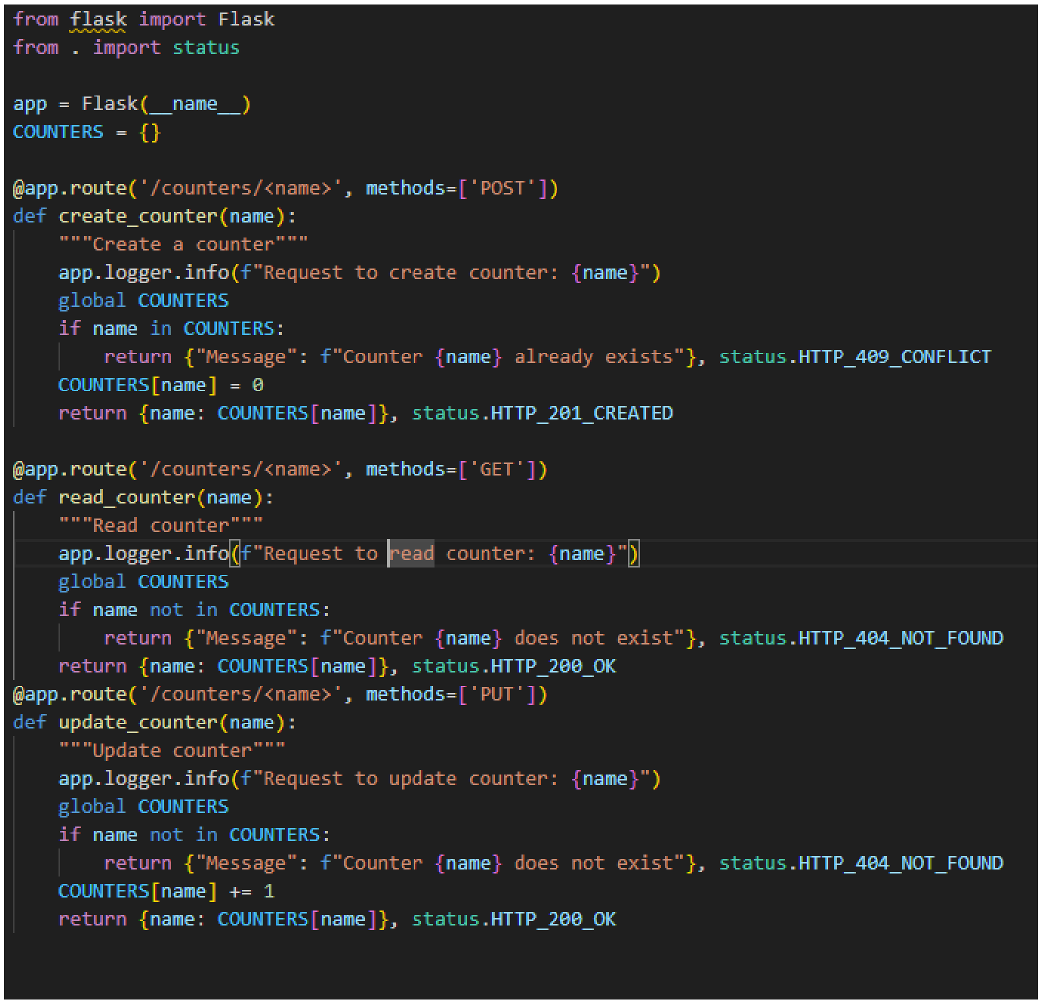
```
import pytest
from src.counter import app, create_counter
from src import status

@pytest.fixture()
def client():
    return app.test_client()

@pytest.mark.usefixtures("client")
class TestCounterEndPoints:

    def test_duplicate_a_counter(self, client): #
        result = client.post('/counters/bar')
        assert result.status_code == status.HTTP_201_CREATED
        result = client.post('/counters/bar')
        assert result.status_code == status.HTTP_409_CONFLICT
    def test_create_a_counter(self, client): #sn
        result = client.post('/counters/foo')
        assert result.status_code == status.HTTP_201_CREATED

    def test_to_counter(self, client):
        re = client.post('/counters/i')
        assert re.status_code == status.HTTP_201_CREATED
        a = client.get('/counters/i')
        assert a.status_code == status.HTTP_200_OK
        base = a.get_json().get("i")
        result = client.put('/counters/i')
        assert result.status_code == status.HTTP_200_OK
        x = client.get('/counters/i')
        assert x.status_code == status.HTTP_200_OK
        update = x.get_json().get("i")
        assert update == base + 1

    def test_to_no_counter(self, client):
        re = client.get('/counters/missed')
        assert re.status_code == status.HTTP_404_NOT_FOUND
    def test_next_misscount(self, client):
        re = client.put('/counters/missed')
        assert re.status_code == status.HTTP_404_NOT_FOUND

    def test_to_next_counter(self, client):
        re = client.post('/counters/ex')
        assert re.status_code == status.HTTP_201_CREATED
        NextRe = client.get('/counters/ex')
        assert NextRe.status_code == status.HTTP_200_OK
        va = NextRe.get_json().get("ex")
        assert va == 0
```

Figure 14 :Test_counter

I had issues running the pytest but had to change the input to the terminal as “python3 -m pytest”

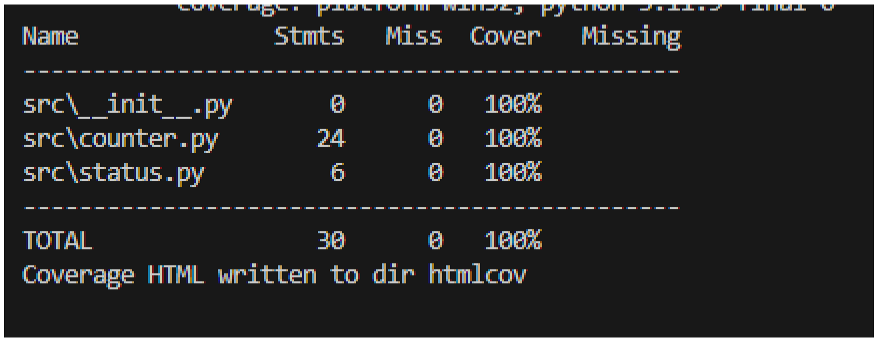
Writing the test_counbter.py file was simple started simple, by first creating it in the src directory. Then by implementing the code found below to import task and initialize task. I did get an error, but the overall output did match. Running the first test case did give me multiple errors including the red “AssertionError: 404 !=201”. So I had to create and import a source code from. import status and a code for the counters. This resulted in a green output. Then for the refactor phase, I had to create a counter bar called “test_duplicate_a_counter” that took in the self and client variables.

This did result in a red but the counter did have to get updated and another refractor case was fixed by using a snippet and global counters. The counter test was used to check the cycle and make sure things were updating correctly from “test_to_counter”. The test to no counter is by checking if the counter is missing using the “test_to_no_counter” method. The next is to check if the update does not exist or that it is missed, meaning that it is not created. The “test_to_next_counter” checks the counter that is created and taken



```python
from flask import Flask
from . import status

app = Flask(__name__)
COUNTERS = {}

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

Figure 15: Counte.py files



```
Name                Stmts   Miss  Cover   Missing
-------------------------------------------------
src\__init__.py         0      0   100%
src\counter.py         24      0   100%
src\status.py           6      0   100%
-------------------------------------------------
TOTAL                  30      0   100%
Coverage HTML written to dir htmlcov
```

Figure 16: Coverage results