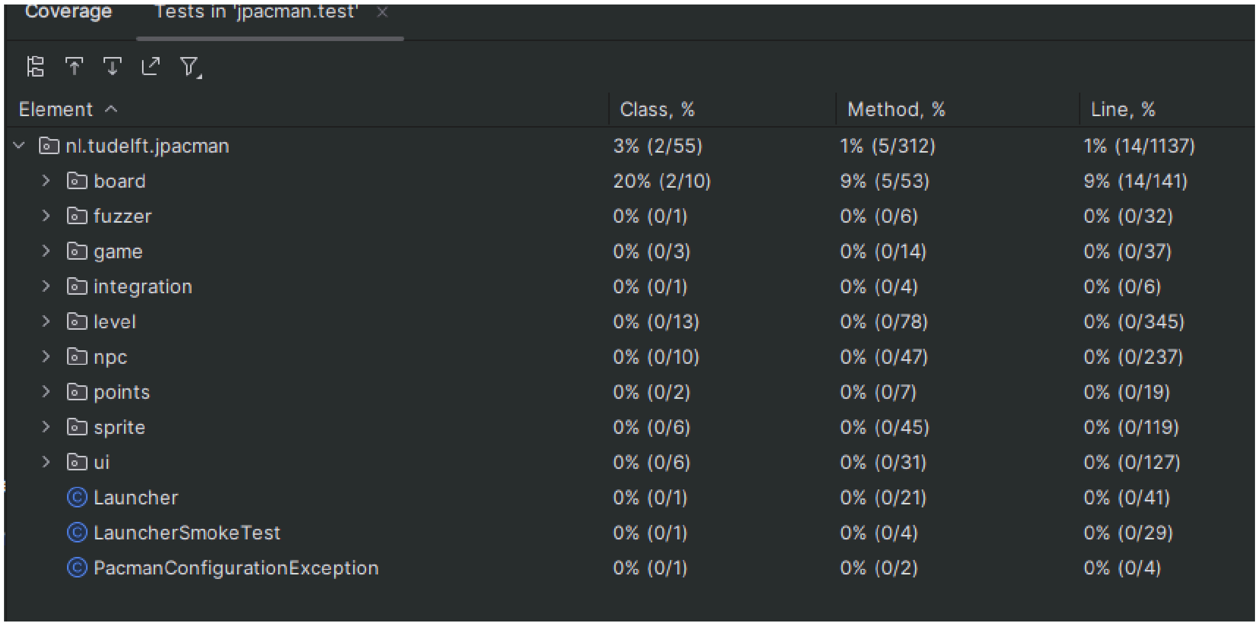
Before we begin, here are all forks links
https://github.com/koyomi7/jpacman
https://github.com/koyomi7/test_coverage/tree/main
https://github.com/koyomi7/tdd/tree/main

Part 1

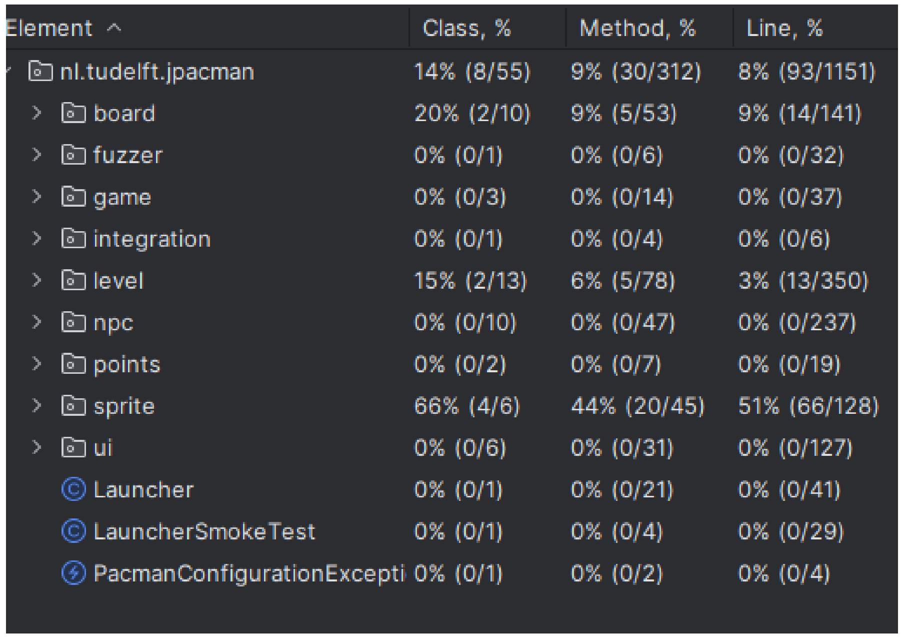

Coverage   Tests in 'jpacman.test'

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)

Question:
Is the coverage good enough?

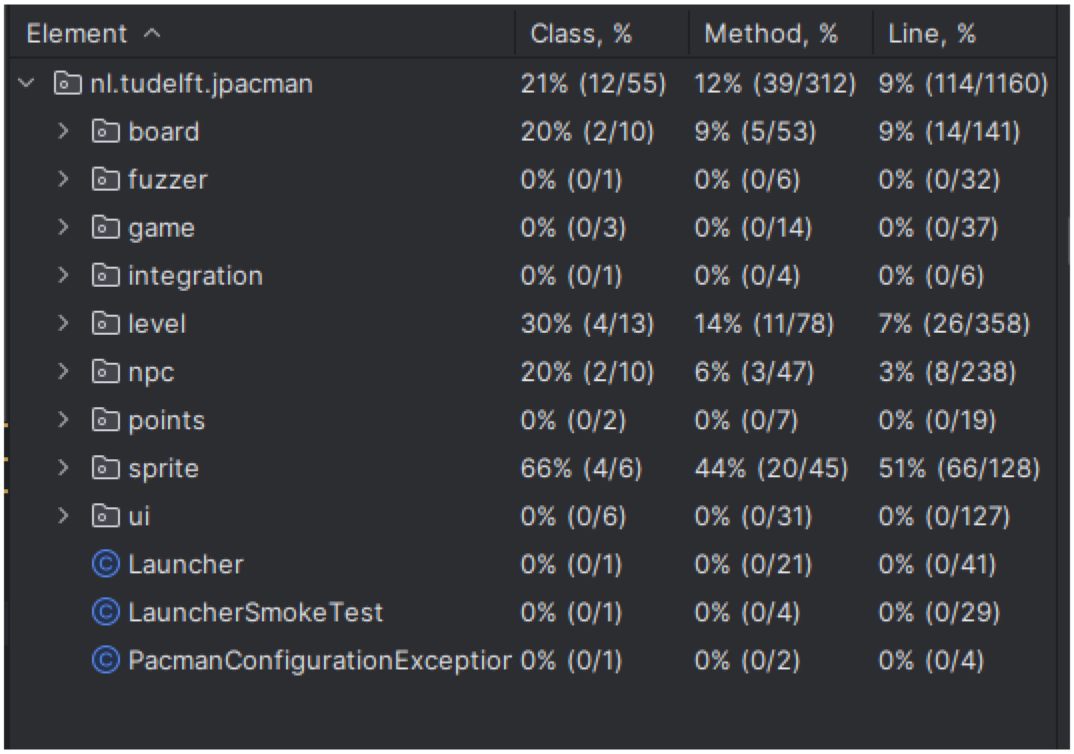
- **As seen on the screenshot, the coverage is not nearly good enough.**

After adding PlayerTest to check if the player is alive, coverage got better

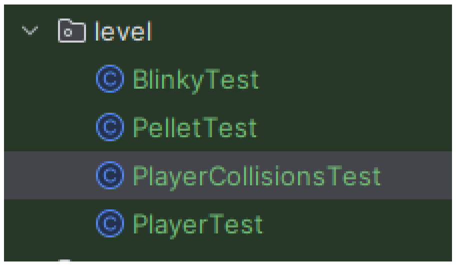


Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	14% (8/55)	9% (30/312)	8% (93/1151)
board	20% (2/10)	9% (5/53)	9% (14/141)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)
game	0% (0/3)	0% (0/14)	0% (0/37)
integration	0% (0/1)	0% (0/4)	0% (0/6)
level	15% (2/13)	6% (5/78)	3% (13/350)
npc	0% (0/10)	0% (0/47)	0% (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)
PacmanConfigurationExcepti	0% (0/1)	0% (0/2)	0% (0/4)

Part 2.1



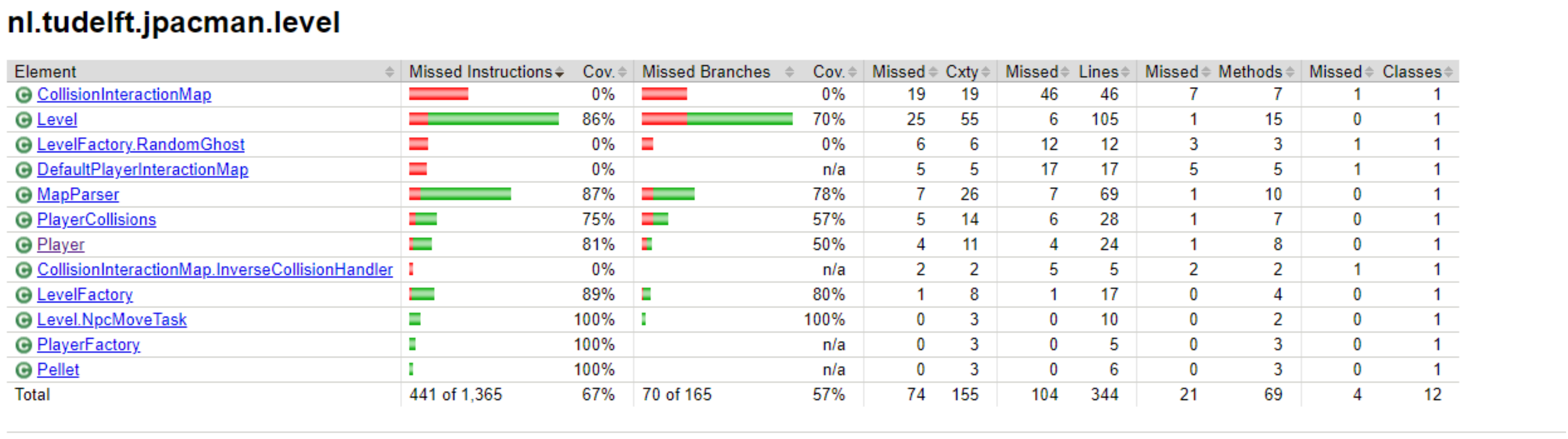
Element	Class, %	Method, %	Line, %
nl.tudelft.jpacman	21% (12/55)	12% (39/312)	9% (114/1160)
board	20% (2/10)	9% (5/53)	9% (14/141)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)
game	0% (0/3)	0% (0/14)	0% (0/37)
integration	0% (0/1)	0% (0/4)	0% (0/6)
level	30% (4/13)	14% (11/78)	7% (26/358)
npc	20% (2/10)	6% (3/47)	3% (8/238)
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)
PacmanConfigurationExceptior	0% (0/1)	0% (0/2)	0% (0/4)



level
- BlinkyTest
- PelletTest
- PlayerCollisionsTest
- PlayerTest

The above is after writing three more methods’ unit tests.

Part 3



nl.tudelft.jpacman.level

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
CollisionInteractionMap		0%		0%	19	19	46	46	7	7	1	1
Level		86%		70%	25	55	6	105	1	15	0	1
LevelFactory.RandomGhost		0%		0%	6	6	12	12	3	3	1	1
DefaultPlayerInteractionMap		0%		n/a	5	5	17	17	5	5	1	1
MapParser		87%		78%	7	26	7	69	1	10	0	1
PlayerCollisions		75%		57%	5	14	6	28	1	7	0	1
Player		81%		50%	4	11	4	24	1	8	0	1
CollisionInteractionMap.InverseCollisionHandler		0%		n/a	2	2	5	5	2	2	1	1
LevelFactory		89%		80%	1	8	1	17	0	4	0	1
Level.NpcMoveTask		100%		100%	0	3	0	10	0	2	0	1
PlayerFactory		100%		n/a	0	3	0	5	0	3	0	1
Pellet		100%		n/a	0	3	0	6	0	3	0	1
Total	441 of 1,365	67%	70 of 165	57%	74	155	104	344	21	69	4	12

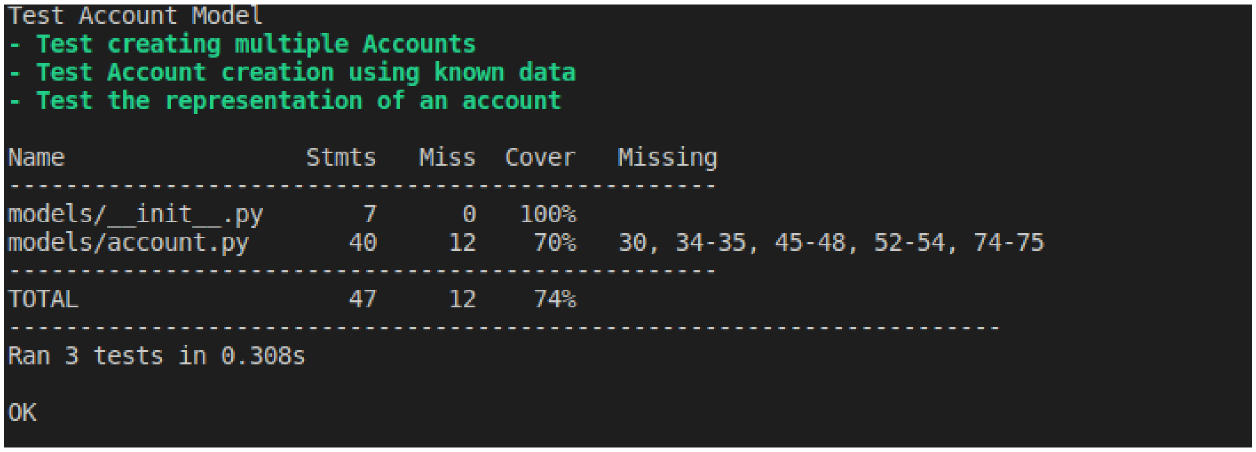
Questions:

- Are the coverage results from JaCoCo similar to the ones you got from IntelliJ in the last task? Why so or why not?
    - **There are similar coverage percentages and highlighted lines of code. Though there are differences in details, I think it is because JaCoCo specifies the instructions for all elements and checks if they were tested, while IntelliJ tells the difference when compiling.**
- Did you find helpful the source code visualization from JaCoCo on uncovered branches?
    - **Yes, it visually tells and specifies what functions have been tested and have not.**
- Which visualization did you prefer and why? IntelliJ's coverage window or JaCoCo's report?
    - **I prefer JaCoCo's visualization because it tells the unchecked targets.**

JPacMan fork link
https://github.com/koyomi7/jpacman

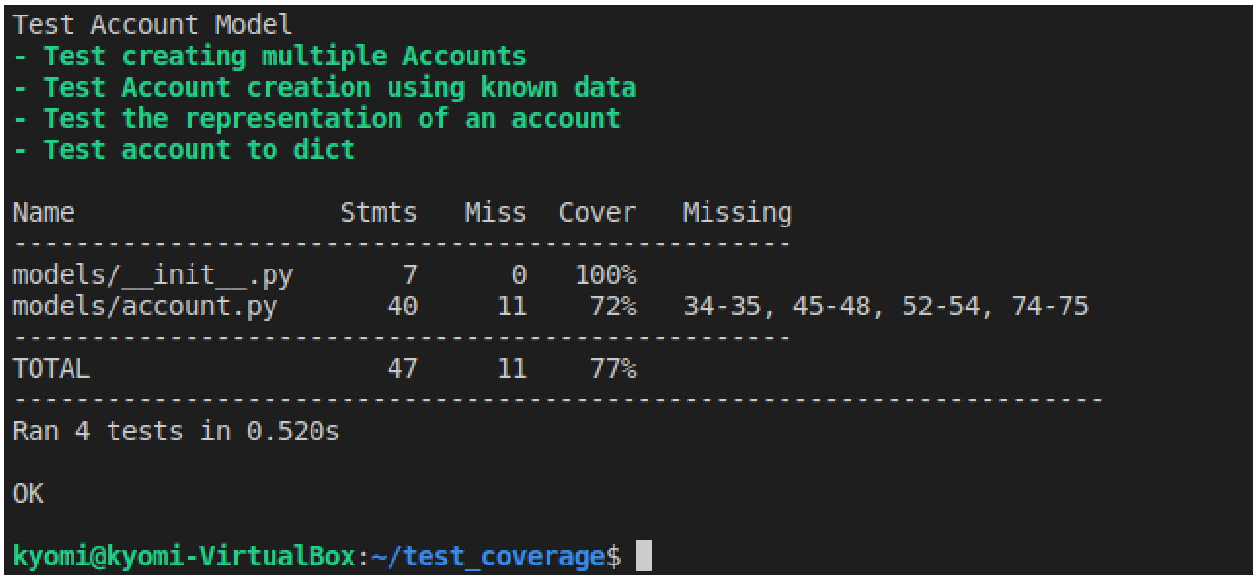
Part 4

Due to issues with the package on Windows, I continued the rest of the lab on Linux.



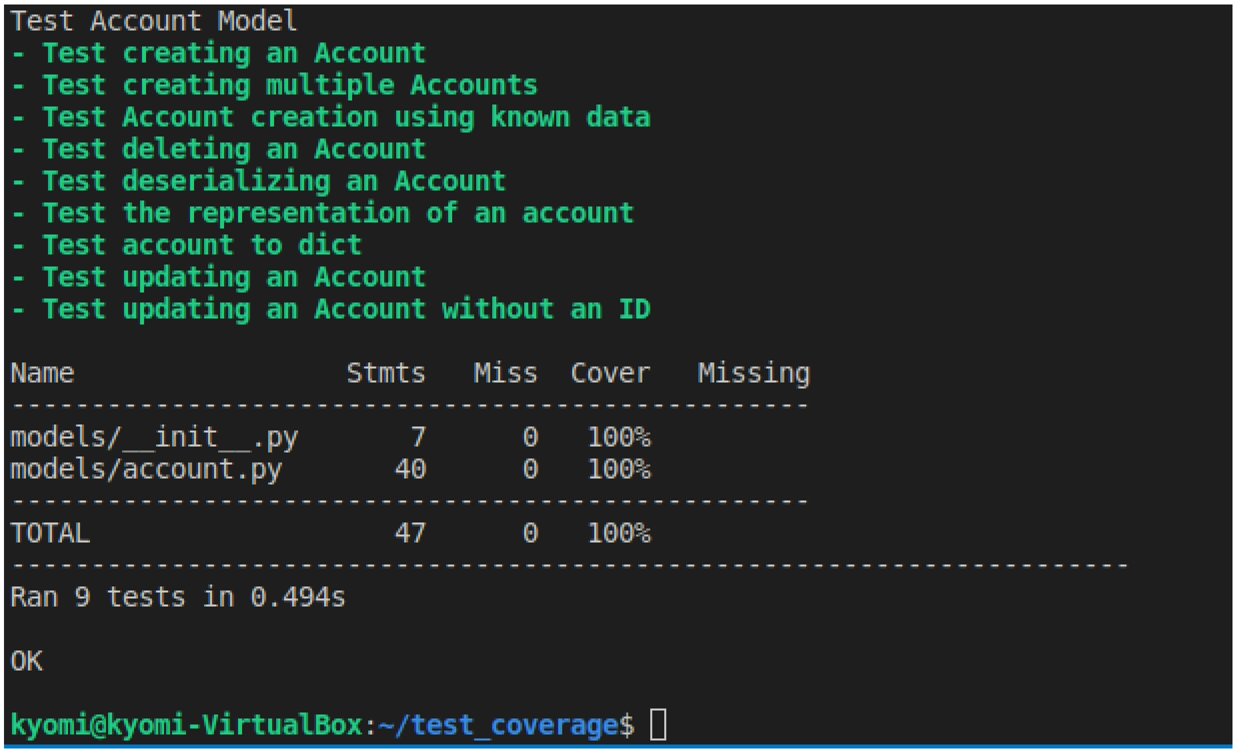
```
Test Account Model
- Test creating multiple Accounts
- Test Account creation using known data
- Test the representation of an account

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 0.308s

OK
```

This is after adding the test case for "__repr__"



```
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 0.520s

OK

kyomi@kyomi-VirtualBox:~/test_coverage$ 
```

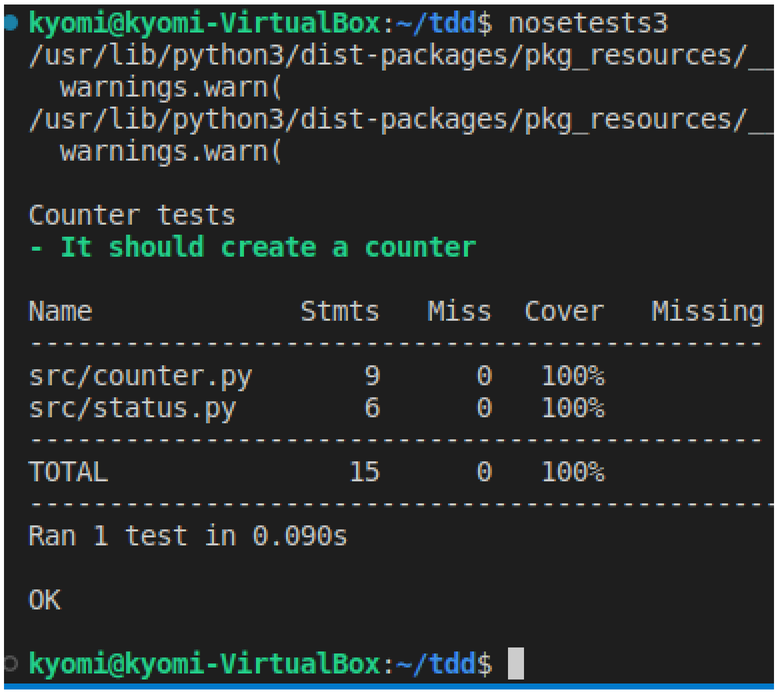
This is after adding the test case for "to_dict"

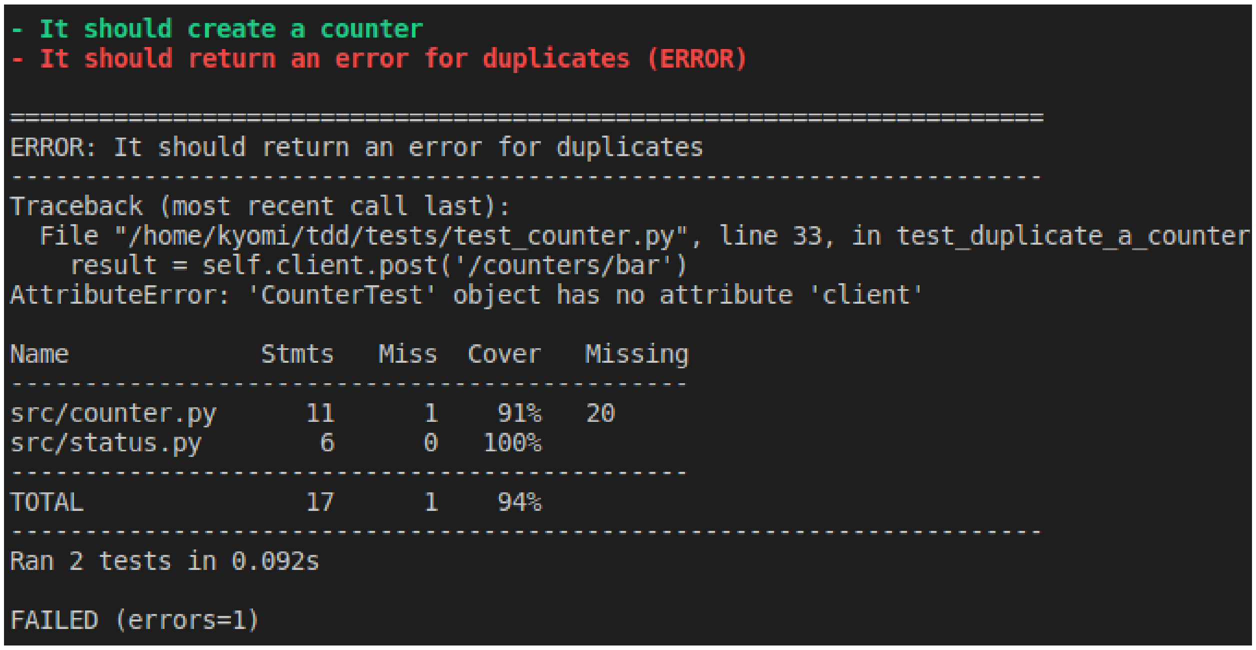


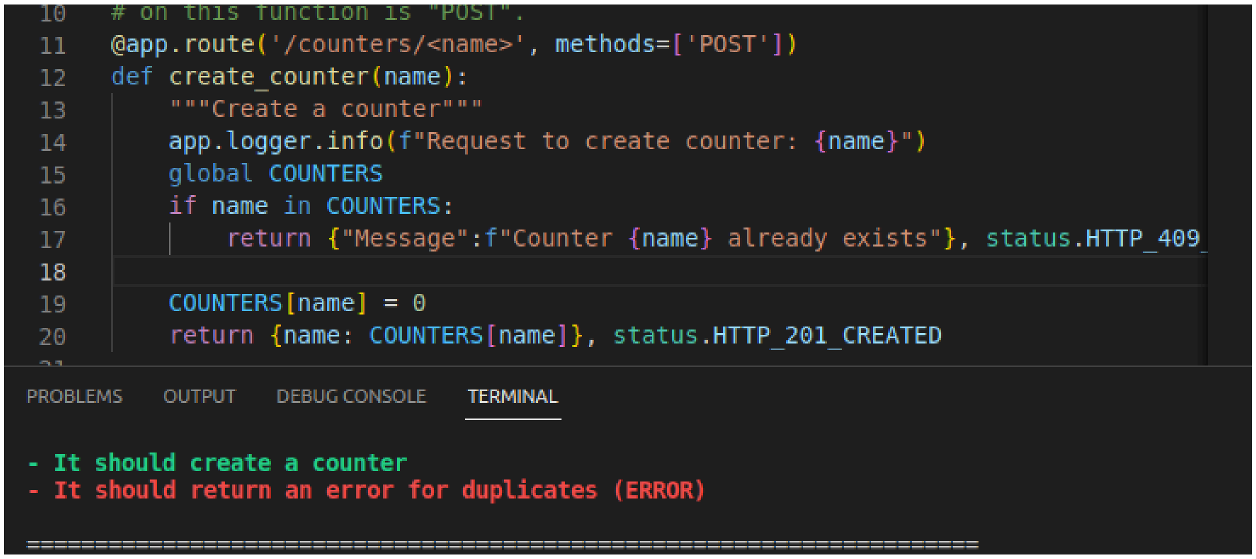
```
Test Account Model
- Test creating an Account
- Test creating multiple Accounts
- Test Account creation using known data
- Test deleting an Account
- Test deserializing an Account
- Test the representation of an account
- Test account to dict
- Test updating an Account
- Test updating an Account without an ID

Name                    Stmts   Miss  Cover   Missing
-----------------------------------------------------
models/__init__.py          7      0   100%
models/account.py          40      0   100%
-----------------------------------------------------
TOTAL                      47      0   100%
----------------------------------------------------------------------
Ran 9 tests in 0.494s

OK

kyomi@kyomi-VirtualBox:~/test_coverage$ 
```

Here is after filling in all test cases.
Link to the fork
https://github.com/koyomi7/test_coverage/tree/main

Part 5


```
kyomi@kyomi-VirtualBox:~/tdd$ nosetests3
/usr/lib/python3/dist-packages/pkg_resources/__
  warnings.warn(
/usr/lib/python3/dist-packages/pkg_resources/__
  warnings.warn(

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 0.090s

OK

kyomi@kyomi-VirtualBox:~/tdd$
```
Shows green after setting up counter.py


```
- It should create a counter
- It should return an error for duplicates (ERROR)

======================================================================
ERROR: It should return an error for duplicates
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/kyomi/tdd/tests/test_counter.py", line 33, in test_duplicate_a_counter
    result = self.client.post('/counters/bar')
AttributeError: 'CounterTest' object has no attribute 'client'

Name                 Stmts   Miss  Cover   Missing
---------------------------------------------------
src/counter.py          11      1    91%   20
src/status.py            6      0   100%
---------------------------------------------------
TOTAL                   17      1    94%
----------------------------------------------------------------------
Ran 2 tests in 0.092s

FAILED (errors=1)
```
Shows red after adding SetUp and test_duplicate_a_counter



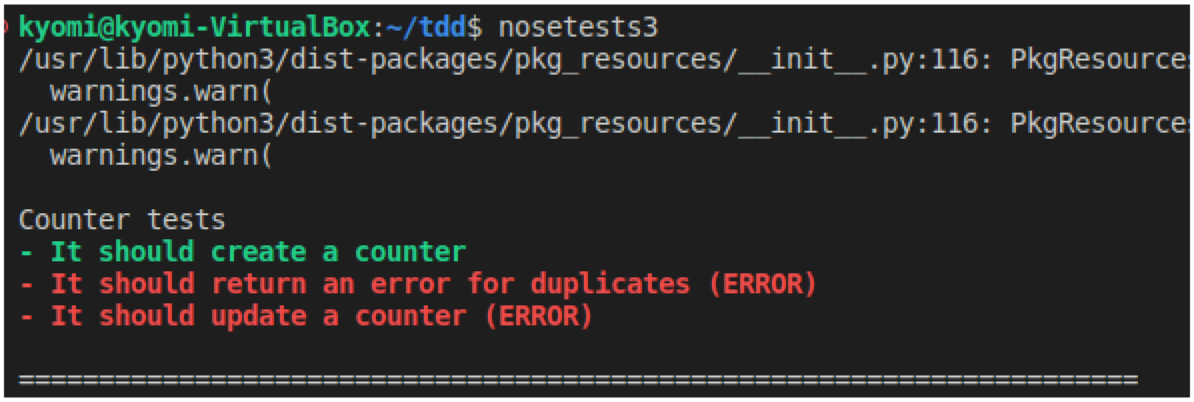
```
# on this function is "POST".
@app.route('/counters/<name>', methods=['POST'])
def create_counter(name):
    """Create a counter"""
    app.logger.info(f"Request to create counter: {name}")
    global COUNTERS
    if name in COUNTERS:
        return {"Message":f"Counter {name} already exists"}, status.HTTP_409_

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
- It should create a counter
- It should return an error for duplicates (ERROR)

======================================================================
```

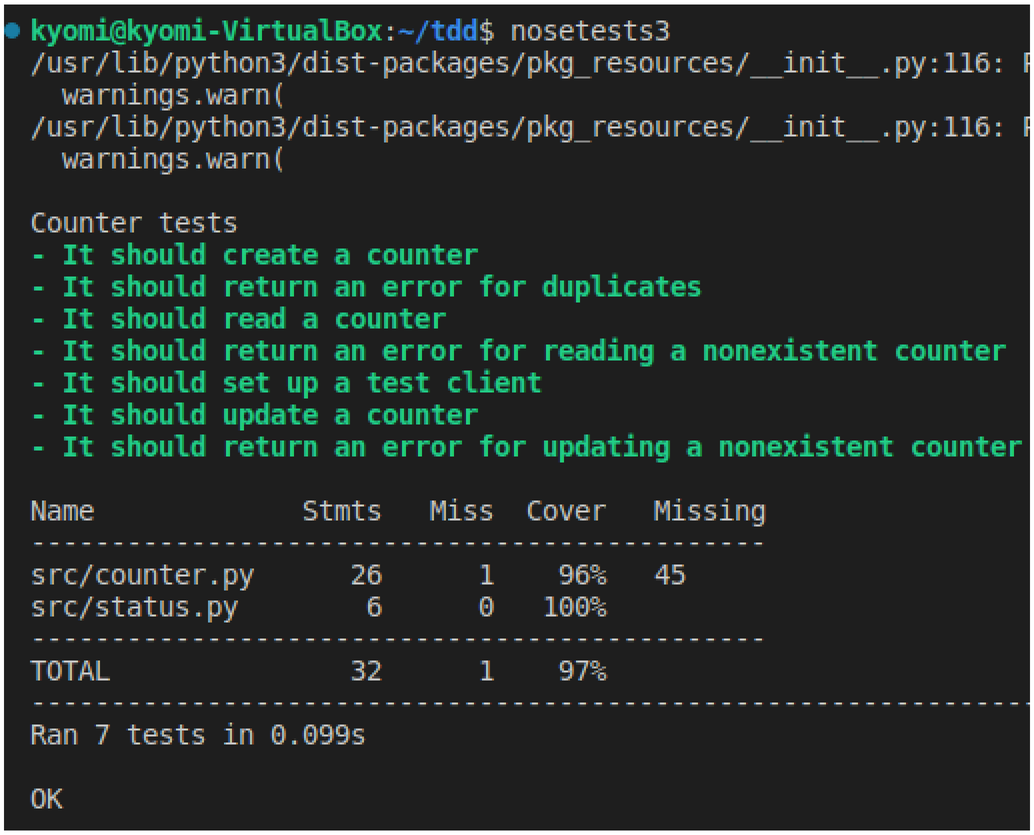
Put
if name **in** COUNTERS:
 return {"Message":f"Counter {name} already exists"}, status.HTTP_409_CONFLICT

Under global COUNTERS, but red still exists.



```
kyomi@kyomi-VirtualBox:~/tdd$ nosetests3
/usr/lib/python3/dist-packages/pkg_resources/__init__.py:116: PkgResource
  warnings.warn(
/usr/lib/python3/dist-packages/pkg_resources/__init__.py:116: PkgResource
  warnings.warn(

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

======================================================================
```

In the red phase after creating a test called test_update_a_counter(self)
Now add all test cases according to the counter.py file



```
kyomi@kyomi-VirtualBox:~/tdd$ nosetests3
/usr/lib/python3/dist-packages/pkg_resources/__init__.py:116: P
  warnings.warn(
/usr/lib/python3/dist-packages/pkg_resources/__init__.py:116: P
  warnings.warn(

Counter tests
- It should create a counter
- It should return an error for duplicates
- It should read a counter
- It should return an error for reading a nonexistent counter
- It should set up a test client
- It should update a counter
- It should return an error for updating a nonexistent counter

Name                Stmts   Miss  Cover   Missing
-------------------------------------------------
src/counter.py         26      1    96%   45
src/status.py           6      0   100%
-------------------------------------------------
TOTAL                  32      1    97%
-----------------------------------------------------------------
Ran 7 tests in 0.099s

OK
```

The line 45 missing is the setUp function, which is a method provided by the unit test framework and is called before each test case, it should be ignored as it's already been tested for all the other cases.
Finished everything with the green phase.
Tdd fork link
https://github.com/koyomi7/tdd/tree/main