

Executive Report














09/17/24 / CS472 – 1001 / Group 2 / Eric Pettie

My repository : https://github.com/EricPettie/CS472_LABS

Group Repository : https://github.com/uid106/CS472_LABS

Task 1 – Jpacman Test Coverage

This coverage is not enough, it's at 0% for almost all the packages.

Coverage Tests in 'jpacman.test' ×				
Element ^	Class, %	Method, %	Line, %	Branch, %
✓  nl.tudelft.jpacman	3% (2/55)	1% (5/312)	1% (14/1127)	0% (1/521)
>  board	20% (2/10)	9% (5/53)	9% (14/141)	1% (1/96)
>  fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
>  game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/14)
>  integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
>  level	0% (0/13)	0% (0/78)	0% (0/343)	0% (0/167)
>  npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/116)
>  points	0% (0/2)	0% (0/7)	0% (0/19)	0% (0/4)
>  sprite	0% (0/6)	0% (0/45)	0% (0/119)	0% (0/48)
>  ui	0% (0/6)	0% (0/31)	0% (0/123)	0% (0/60)
 Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/6)
 LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
 PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)

Task 2 – Added PlayerTest class

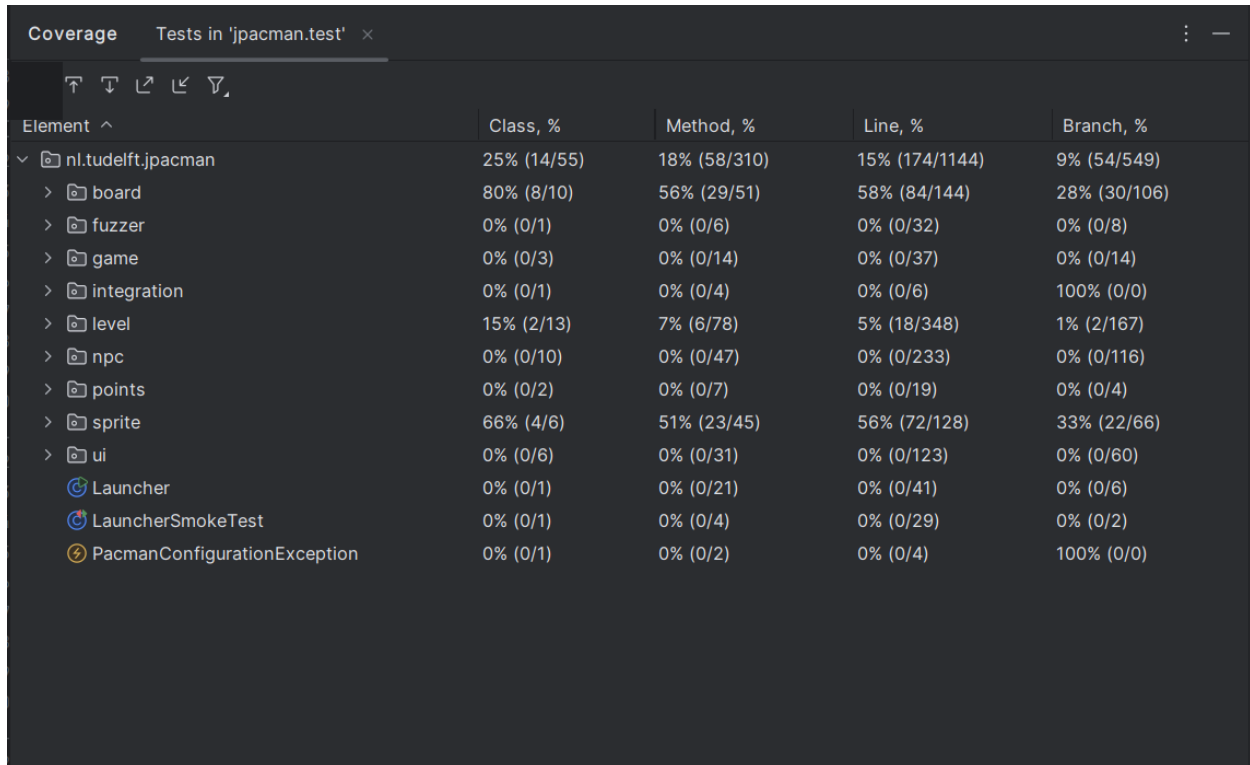
Coverage Tests in 'jpacman.test' ×				
Element ^	Class, %	Method, %	Line, %	Branch, %
✓ nl.tudelft.jpacman	14% (8/55)	10% (32/312)	8% (102/1141)	4% (24/539)
> board	20% (2/10)	9% (5/53)	9% (14/141)	1% (1/96)
> fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
> game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/14)
> 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)
> npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/116)
> 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% (0/60)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/6)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)

After we added the PlayerTest class, the coverage increased for the level

Task 2.1 – Added BoardFactoryTest

This test validated the functionality of the BoardFactory class, making sure that the creation of board, ground, and wall objects work correctly.

As you can see, the test coverage increased for the board



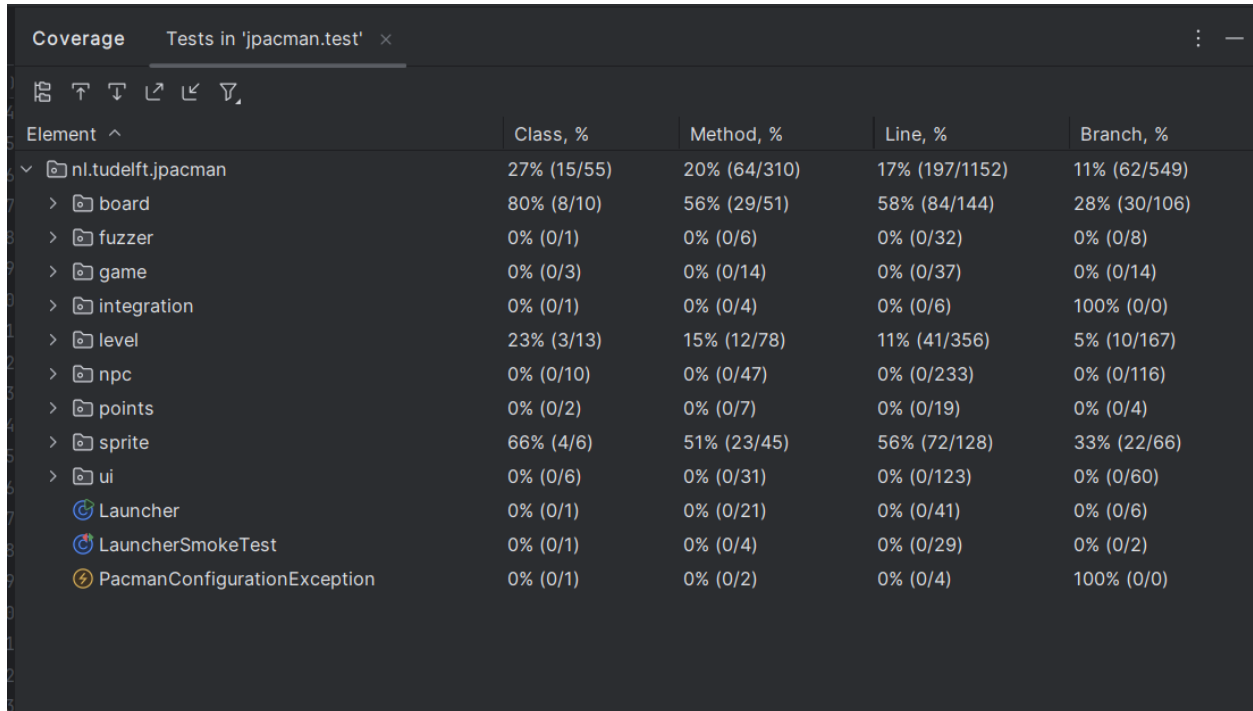
The screenshot shows the Coverage tool in IntelliJ IDEA. The title bar indicates 'Coverage' and 'Tests in 'jpacman.test'' with a close button. Below the title bar is a toolbar with icons for sorting and filtering. The main table displays coverage data for various elements. The 'Element' column lists packages and classes, while the other columns show 'Class, %', 'Method, %', 'Line, %', and 'Branch, %' with their respective counts in parentheses.

Element ^	Class, %	Method, %	Line, %	Branch, %
▼ nl.tudelft.jpacman	25% (14/55)	18% (58/310)	15% (174/1144)	9% (54/549)
> board	80% (8/10)	56% (29/51)	58% (84/144)	28% (30/106)
> fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
> game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/14)
> 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)
> npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/116)
> points	0% (0/2)	0% (0/7)	0% (0/19)	0% (0/4)
> sprite	66% (4/6)	51% (23/45)	56% (72/128)	33% (22/66)
> ui	0% (0/6)	0% (0/31)	0% (0/123)	0% (0/60)
⌚ Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/6)
⌚ LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
⚡ PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)

Added PlayerCollisionsTest

This test made sure collision logic was working, it ensured that when a player collides with a ghost the player dies. If the player consumes a pellet, points are added and the pellet is removed.

Coverage increased for level



The screenshot shows the Coverage tool in IntelliJ IDEA, displaying test results for 'pacman.test'. The table lists various elements and their coverage percentages for Class, Method, Line, and Branch.

Element ^	Class, %	Method, %	Line, %	Branch, %
nl.tudelft.pacman	27% (15/55)	20% (64/310)	17% (197/1152)	11% (62/549)
board	80% (8/10)	56% (29/51)	58% (84/144)	28% (30/106)
fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/14)
integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
level	23% (3/13)	15% (12/78)	11% (41/356)	5% (10/167)
npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/116)
points	0% (0/2)	0% (0/7)	0% (0/19)	0% (0/4)
sprite	66% (4/6)	51% (23/45)	56% (72/128)	33% (22/66)
ui	0% (0/6)	0% (0/31)	0% (0/123)	0% (0/60)
Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/6)
LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)

Added DefaultPointCalculatorTest


This test verifies that no points are added if a player collides with a ghost, it also verifies that the correct amount of points are added when a player consumes a pellet. Also ensures that no points are added when the player moves.

Coverage increased for points from 0% to 50%

Coverage Tests in 'jpacman.test' ×				
⌵ ⌴ ⌶ ⌷ ⌸ ⌹				
Element ^	Class, %	Method, %	Line, %	Branch, %
✓ nl.tudelft.jpacman	29% (16/55)	21% (68/310)	17% (207/1153)	11% (65/549)
> board	80% (8/10)	56% (29/51)	58% (84/144)	28% (30/106)
> fuzzer	0% (0/1)	0% (0/6)	0% (0/32)	0% (0/8)
> game	0% (0/3)	0% (0/14)	0% (0/37)	0% (0/14)
> integration	0% (0/1)	0% (0/4)	0% (0/6)	100% (0/0)
> level	23% (3/13)	15% (12/78)	11% (41/356)	5% (10/167)
> npc	0% (0/10)	0% (0/47)	0% (0/233)	0% (0/116)
> points	50% (1/2)	57% (4/7)	50% (10/20)	75% (3/4)
> sprite	66% (4/6)	51% (23/45)	56% (72/128)	33% (22/66)
> ui	0% (0/6)	0% (0/31)	0% (0/123)	0% (0/60)
⚙ Launcher	0% (0/1)	0% (0/21)	0% (0/41)	0% (0/6)
⚙ LauncherSmokeTest	0% (0/1)	0% (0/4)	0% (0/29)	0% (0/2)
⚡ PacmanConfigurationException	0% (0/1)	0% (0/2)	0% (0/4)	100% (0/0)











Task 3 – JaCoCo report on JPacman

Below is the JaCoCo report found in build/reports/jacoco/test/html/index.html



jpacman

jpacman

Element	Missed Instructions	Cov	Missed Branches	Cov	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
 nl.tudelft.jpacman.level	<div><div></div></div>	68%	<div><div></div></div>	58%	73 155	103 344	21 69	4 12
 nl.tudelft.jpacman.npc.ghost	<div><div></div></div>	71%	<div><div></div></div>	55%	56 105	43 181	5 34	0 8
 nl.tudelft.jpacman.ui	<div><div></div></div>	77%	<div><div></div></div>	47%	54 86	21 144	7 31	0 6
 default	<div><div></div></div>	0%	<div><div></div></div>	0%	12 12	21 21	5 5	1 1
 nl.tudelft.jpacman.board	<div><div></div></div>	86%	<div><div></div></div>	58%	44 93	2 110	0 40	0 7
 nl.tudelft.jpacman.sprite	<div><div></div></div>	86%	<div><div></div></div>	59%	30 70	11 113	5 38	0 5
 nl.tudelft.jpacman	<div><div></div></div>	69%	<div><div></div></div>	25%	12 30	18 52	6 24	1 2
 nl.tudelft.jpacman.points	<div><div></div></div>	60%	<div><div></div></div>	75%	1 11	5 21	0 9	0 2
 nl.tudelft.jpacman.game	<div><div></div></div>	87%	<div><div></div></div>	60%	10 24	4 45	2 14	0 3
 nl.tudelft.jpacman.npc	<div><div></div></div>	100%		n/a	0 4	0 8	0 4	0 1
Total	1,208 of 4,694	74%	292 of 637	54%	292 590	228 1,039	51 268	6 47

- There is similar information covered, but the percentages are slightly different. JaCoCo seems to be more detailed than IntelliJ, which may be the cause.
- Yes I find the source code visualization is helpful, highlighting exactly where tests are missing makes it very easy to find.
- JaCoCo's is a bit easier to navigate and see where coverage is missing than IntelliJ, so I prefer JaCoCo

Task 4 – Python test coverage

Below are the initial results, our goal is to raise coverage to 100%

```
Eric@DESKTOP-NEFIEHJ MINGW64 ~/test_coverage (main)
$ pytest --cov=models
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Users\Eric\tes
t_coverage\venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Eric\test_coverage
configfile: pytest.ini
plugins: cov-5.0.0
collecting ... collected 2 items

tests/test_account.py::test_create_all_accounts PASSED [ 50%]
tests/test_account.py::test_create_an_account PASSED [100%]

----- coverage: platform win32, python 3.12.6-final-0 -----
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%

===== 2 passed in 1.41s =====
(venv)
Eric@DESKTOP-NEFIEHJ MINGW64 ~/test_coverage (main)
```

Below are the results after added the test for line 26

```
$ pytest --cov=models tests/
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0 -- C:\Users\Eric\test_c
venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Eric\test_coverage
configfile: pytest.ini
plugins: cov-5.0.0
collecting ... collected 3 items

tests/test_account.py::test_create_all_accounts PASSED [ 33%]
tests/test_account.py::test_create_an_account PASSED [ 66%]
tests/test_account.py::test_repr PASSED [100%]

----- coverage: platform win32, python 3.12.6-final-0 -----
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%

===== 3 passed in 0.71s =====
(venv)
```

- The definitions that required tests to be made were:

- Update
- Create
- Delete
- All
- Find
- From_Dict

- Below are some of the code snippets for the test cases

```
def test_create_account():
    """Test Account creation"""
    account = Account(name="test created", email="test@example.com")
    account.create()

    assert len(Account.all()) == 1
    created_account = Account.all()[0]
    assert created_account.name == "test created"
    assert created_account.email == "test@example.com"
```

```
def test_find_account():
    """Test Account lookup by ID"""
    account = Account(name="test account find", email="test@example.com")
    account.create()

    found_account = Account.find(account.id)
    assert found_account is not None
    assert found_account.name == "test account find"
```

```
def test_from_dict():
    data = {
        "name": "New Name",
        "email": "test@example.com",
        "phone_number": "123456789",
        "disabled": True
    }
    account = Account()
    account.from_dict(data)

    assert account.name == "New Name"
    assert account.email == "test@example.com"
    assert account.phone_number == "123456789"
    assert account.disabled == True
```

```
def test_update_account():
    """Test Account update"""
    account = Account(name="Test Name", email="test@example.com")
    account.create()

    account.name = "Updated Name"
    account.update()

    updated_account = Account.find(account.id)
    assert updated_account.name == "Updated Name"
```


Below is the screenshot showing 100% coverage:

```
C:\Users\Eric\test_coverage\models\account.py:75: LegacyAPIWarning: The Query.get() method is considered legacy as of the 1.x series of SQLAlchemy and becomes a legacy construct in 2.0. The method is now available as Session.get() (deprecated since: 2.0) (Background on SQLAlchemy 2.0 at: https://sqlalche.me/e/b8d9)
    return cls.query.get(account_id)

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html

----- coverage: platform win32, python 3.12.6-final-0 -----
Name                               Stmts  Miss  Cover   Missing
-----
models\__init__.py                   7      0   100%
models\account.py                   40      0   100%
-----
TOTAL                               47      0   100%
```

Task 5 - TDD

For this task we are implementing test cases which we will use to create definitions to pass those test cases we created.

```
def test_create_a_counter():

    client = app.test_client()
    result = client.post('/counters/goo')
    assert result.status_code == status.HTTP_201_CREATED
```

After running this code we see an error message because of the function not being implemented yet

These are the implemented functions in counter.py:

```
@app.route('/counters/<name>', methods=['POST'])
def create_counter(name):
    """Creating the 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):
    """Updating the counter by incrementing by 1"""
    app.logger.info(f"Request to update counter: {name}")
    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
```

After reviewing the code for improvements:

```
----- coverage: platform win32, python 3.12.6-final-0 -----
Name                               Stmts  Miss  Cover   Missing
-----
src\__init__.py                     0      0   100%
src\counter.py                     24      9    62%   23-27, 33-36
src\status.py                       6      0   100%
-----
TOTAL                               30      9    70%
```

I see that lines 23-27 and 33-36 were missing, I added the new test cases which put me to 96% coverage

```
def test_read_non_existent_counter(self, client):
    """should return 404 for the non-existent counter"""
    result = client.get('/counters/does_not_exist')
    assert result.status_code == status.HTTP_404_NOT_FOUND

def test_update_non_existent_counter(self, client):
    """should return 404 for updating a non-existent counter"""
    result = client.put('/counters/does_not_exist')
    assert result.status_code == status.HTTP_404_NOT_FOUND
```

```
----- coverage: platform win32, python 3.12.6-final-0 -----
Name                Stmts  Miss  Cover   Missing
-----
src\__init__.py      0      0   100%
src\counter.py       24      1    96%    36
src\status.py        6      0   100%
-----
TOTAL                30      1    97%
```