

ITCS473 Exercise JPacman (1)

Submission: MyCourses

Type of submission: Individual

Due date: 14 August 2025 at 23:55

Deliverable: This completed lab sheet (PDF).

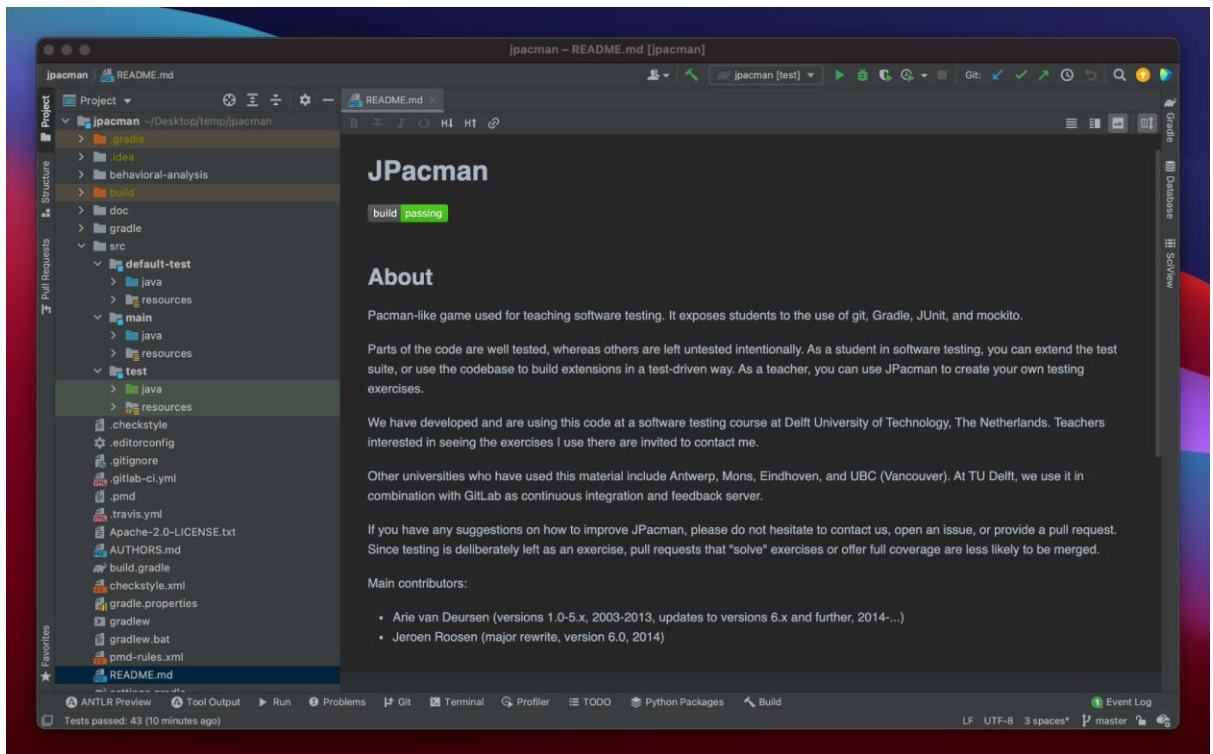
Overview:

You will apply the different testing techniques we teach to a simple game called JPACMAN, inspired by Pac-Man and written in Java. The amount of coding that needs to be done is relatively small: the focus is on testing.

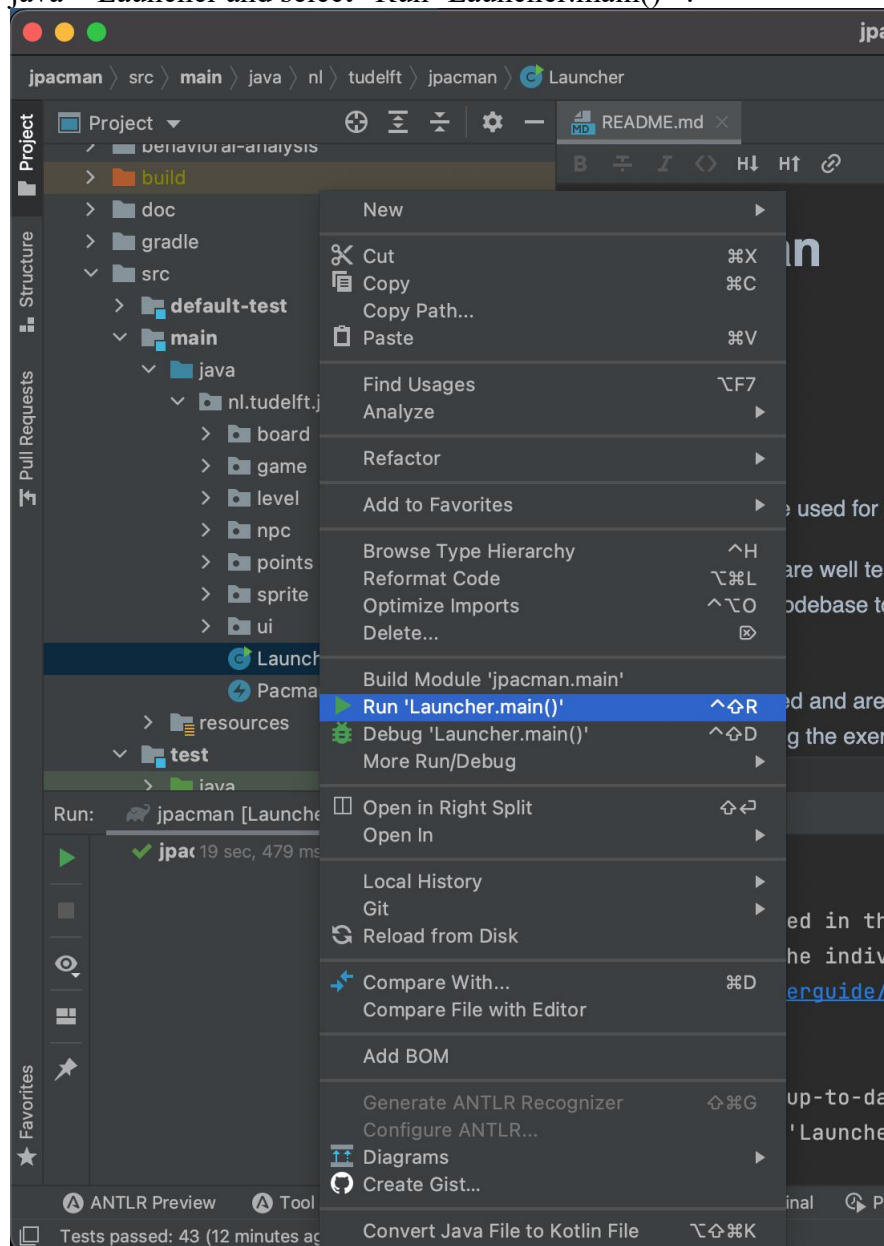
This JPacman exercise is adapted from Software Quality and Testing SQT Labwork, CSE1110 Edition 2018/2019 by Arie van Deursen, Mauricio Aniche Casper Boone, Max Lopes Cunha, Azqa Nadeem, Delft University of Technology

Instructions:

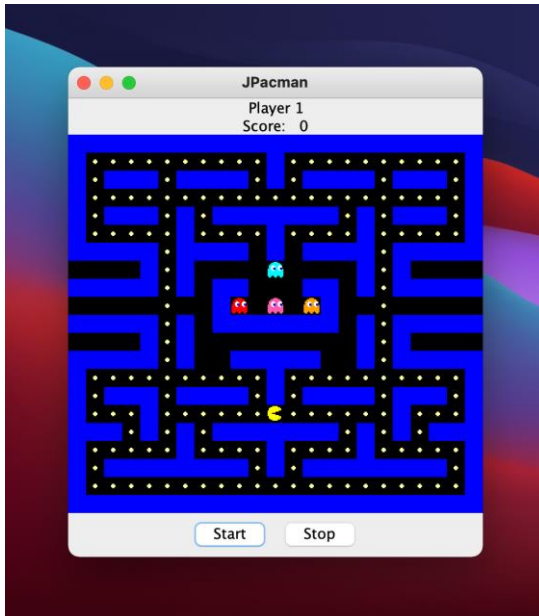
1. Make a copy of this file (File > Download).
2. Fork the JPacman repository from: <https://github.com/MUICT-SERU/jpacman> to your GitHub account.
3. From your GitHub account, clone the forked JPacman repository to your local machine.
4. Open IntelliJ and open the cloned JPacman project. Wait for a while until all the required dependencies are downloaded. You should see the project structure like the photo below.



5. After you have everything set up, try to run and play the game by going to main > java > Launcher and select “Run ‘Launcher.main()’”.



6. You should see the GUI of the JPacman game show up like below.

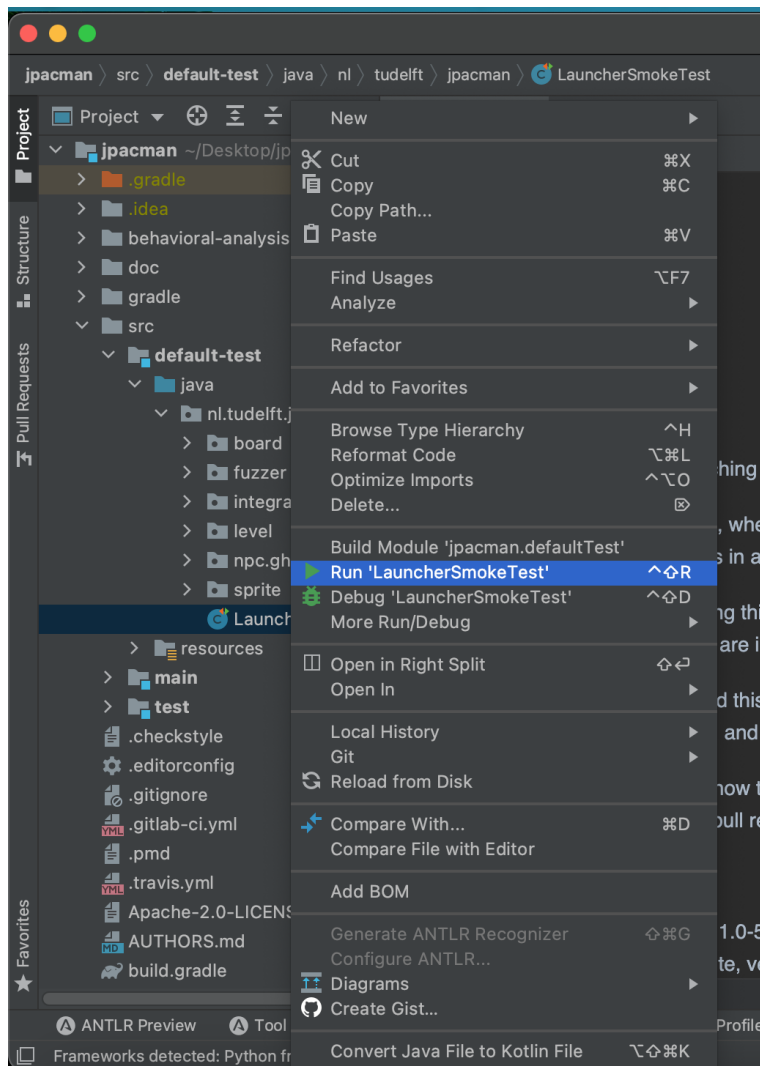


7. Play the game to understand how it works.
8. You can also explore the game structure, build system (Gradle), and the available tests (in default-test, and test folders).
9. Put the URL of your forked repository here:

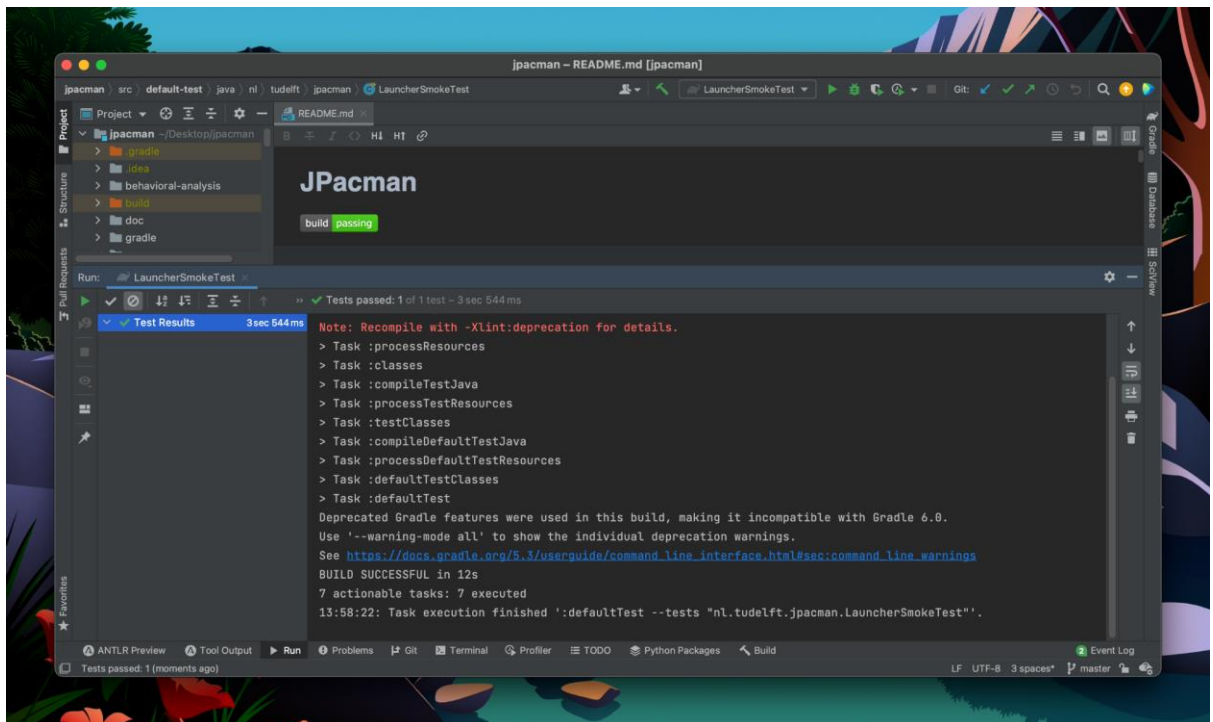
<https://github.com/Mahumeaw/jpacman.git>

10. Next, we will try running the smoke test to make sure that everything is working okay. Open the `LauncherSmokeTest` class in `src/default-test/java/nl.tudelft.jpacman`. Then, right-click and select *Run 'LauncherSmokeTest'*. You will see the JPacman showing up and closing.

ITCS473 SW Quality Assurance and Testing
Exercise



11. You should see the result like this.



12. Studying the code in the LauncherSmokeTest.java file and performing some research about the concept of smoke test and assert statements in Java. Answer the questions below.

a. Why do we need to do a smoke test?

Smoke test is performed the quickly in important function to assert that including start, get points, moving, and move to monsters.

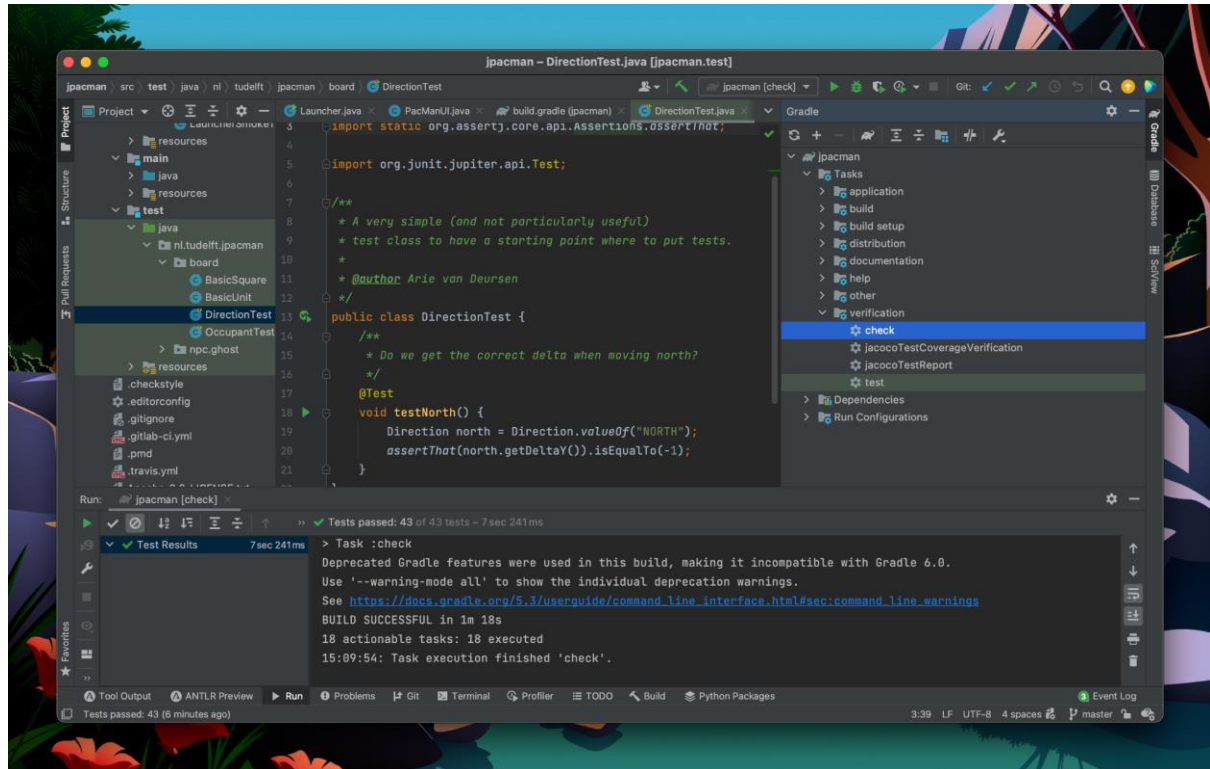
b. What kind of checks are performed in this smoke test?

Assertion-based testing

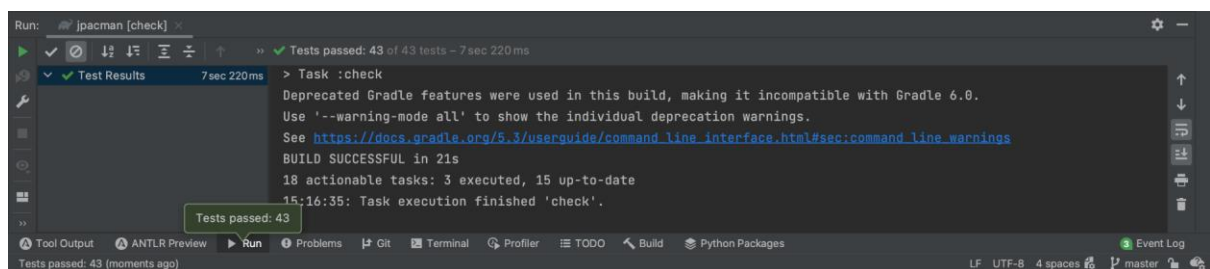
13. Let's try to make a small change and run the smoke test again. Open the PacManUI class in nl.tudelft.tudelft.jpacman.ui package. Change the code in the constructor to be as follows.

```
public PacManUI(final Game game, final Map<String, Action> buttons,
                final Map<Integer, Action> keyMappings,
                ScoreFormatter scoreFormatter) {
    super("JPacman 2024");
    assert game != null;
    assert buttons != null;
    assert keyMappings != null;
```

14. Launch the LauncherSmokeTest again and see the title changes (need to quickly see it!).
15. Try to run the existing tests by opening the Gradle pane. Then, select jpacman > Tasks > verification > check.



16. Wait until the test executions have finished. You will see the result that 43 tests passed.



17. Study Gradle and how the Gradle check command works.

“Gradle is a build automation tool for multi-language software development. It controls the development process in the tasks of compilation and packaging to testing, deployment, and publishing. Supported languages include Java, C/C++, and JavaScript. -- Wikipedia”

We'll use Gradle for testing the JPacman project. To learn Gradle, which is the widely used building tool nowadays, you can take the tutorial here:

<https://www.linkedin.com/learning/learning-gradle>

18. Capture the screen of your version of the JPacman project after running the Gradle check command and paste it below.

```
✓ 43 tests passed 43 tests total, 1sec 887ms

> Task :compileJava UP-TO-DATE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE
> Task :compileTestJava UP-TO-DATE
> Task :processTestResources UP-TO-DATE
> Task :testClasses UP-TO-DATE
> Task :compileDefaultTestJava UP-TO-DATE
> Task :processDefaultTestResources UP-TO-DATE
> Task :defaultTestClasses UP-TO-DATE
> Task :checkstyleDefaultTest
> Task :checkstyleMain
> Task :checkstyleTest
> Task :pmdDefaultTest
This analysis could be faster, please consider using Incremental Analysis: https://pmd.github.io/latest/faq.html#faster
> Task :pmdMain
This analysis could be faster, please consider using Incremental Analysis: https://pmd.github.io/latest/faq.html#faster
> Task :pmdTest
This analysis could be faster, please consider using Incremental Analysis: https://pmd.github.io/latest/faq.html#faster
> Task :spotbugsDefaultTest
> Task :spotbugsMain
> Task :spotbugsTest
> Task :defaultTest
> Task :test
> Task :jacocoTestReport
> Task :check
```

19. Save this file as a PDF and submit it to MyCourses.