INF1-IP

Lecture 9 – Code quality 1: Testing and documentation

Rayo Verweij

19/10/2026

THE UNIVERSITY of EDINBURGH
informatics

> Coding best practices

- **Plan** the steps of your algorithm using **pseudocode**
- **Decompose** problems into smaller, more manageable chunks

- Give your variables clear, **descriptive names**
  - Use capital letters for class names

- Add **comments** to explain non-trivial steps
  - Use triple-quoted **docstrings** to add documentation to functions

- Use consistent **indentation**

- **DRY: Don't Repeat Yourself** – refactor out similar pieces of code.

> Automated testing

Assertions allow us to verify that code is running correctly.

If they evaluate to `True`, nothing happens; if `False`, they throw an error.

```
assert 2 + 2 == 4
print("One assertion has passed")
```

Assertions can be given a string that will be displayed if it fails.



```
list = [1, 2, 3]
assert list[1] == 1, "Lists are zero-indexed!"    # Throws AssertionError
```

```
→  AssertionError: Lists are zero-indexed!
```

> Test-driven development

Test-driven development is a slow, methodical approach to coding in which you **write tests before code**.

It follows a three-part structure:

- 🔴 **Red:** Write a test. It fails, as you don't have code yet.
- 🟢 **Green:** Write the minimum code necessary for the test to pass.
- 🔀 **Refactor:** When necessary, re-organise your code for legibility.

❗ Preparing for the Class Test

🐍 **Work through the self-guided notebooks.**

They're relevant, I wrote them.

📚 **Check "Further reading and practice" for alternative explanations.**

Note – they cover the material in a different order to us.

👩‍💻 **Drop in to InfPALS and/or post on EdStem with any questions.**

For details, see https://infpals.github.io

🍜 **Eat well, get some sleep, and take care of yourself!**