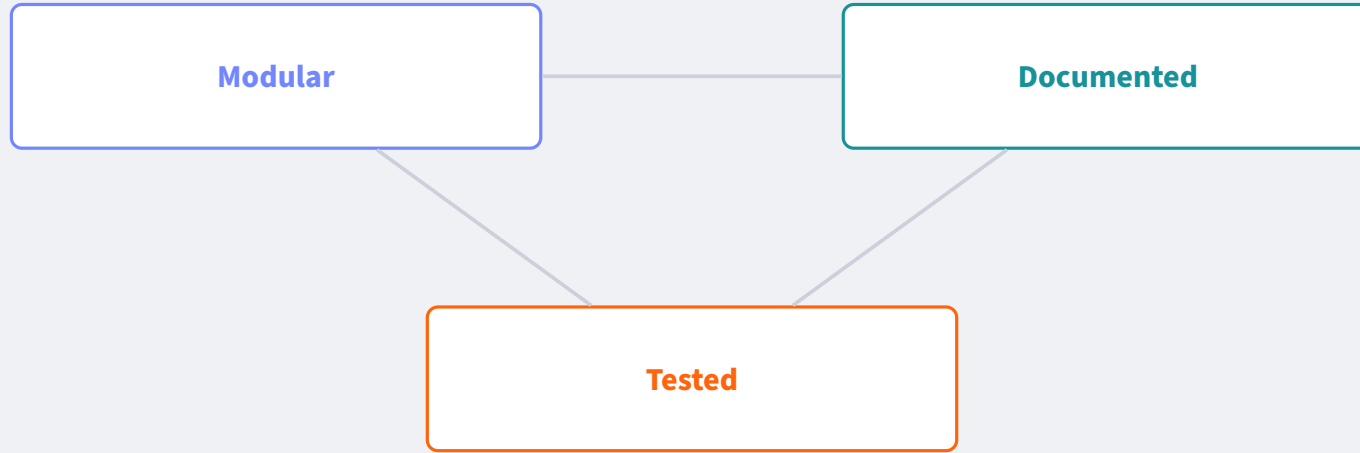


Testing

Week 10. The last meeting week.





Each one makes the other two easier. Together they are **50%** of the project mark.

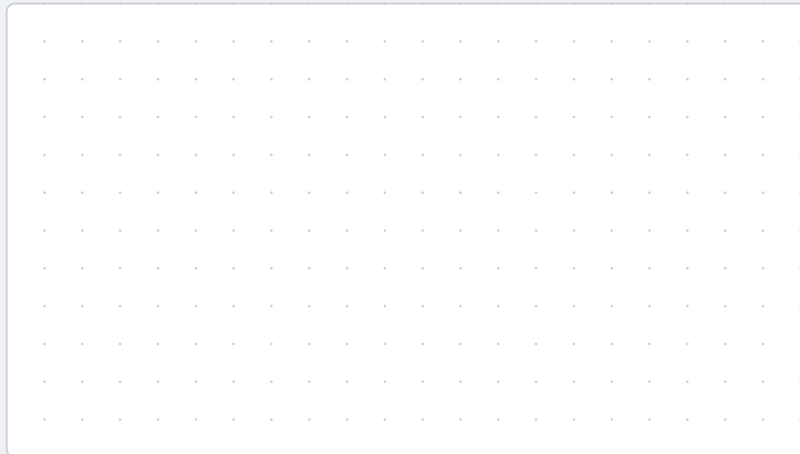
```
assert add_one(5) == 6, "The function gave the wrong answer."
```

Two things come after the word assert. What is each one for?

A claim that must be true, and the sentence you will read at three in the morning when it is not.

```
def feedback(guess, chosen):  
    if guess < chosen:  
        return "too low"  
    if guess > chosen:  
        return "too high"  
    return "correct"
```

One test passes. Is the function tested?



Count the things the function can do, not the number of functions. This one does three.

Zero

One

Empty

Negative

Pick one function in your library. What does it do with each of these?

The top band asks for edge cases the tutorial does not exercise. These four are where they live.

```
import random
random.seed(0)
assert roll_a_dice() == 4
random.seed(1)
assert roll_a_dice() == 2
```

The function gives a different answer every time. So what makes this test pass, every time?

Seeding fixes the randomness without removing it. You can also assert the shape of a thousand rolls rather than one.

```
# test_tsp.py
import tsp
def test_get_cost():
    """Check the cost of a tour we worked out by hand."""
    assert tsp.get_cost(tour=[0, 1, 2, 0], matrix=m) == 24
test_get_cost()
```

Then, in a terminal:

```
python test_tsp.py
```

A separate file, one function per behaviour, each with a docstring saying what it checks. No library needed.

```
>>> import tsp
>>> tsp.get_tour(number_of_stops=5)
[0, 1, 2, 3, 4, 0]
```

```
python -m doctest README.md
```

You already have a test file. So why write these as well?

A doctest checks that your documentation is still true. It is not where you test the mathematics.

Between you, list every function in your library. Put a cross against each one with no test.

This is the question to take into the meeting this week. It turns into actions better than anything else on the agenda.

☐ main.tex

☐ main.pdf

☐ references.bib

☐ <library>.py

☐ test_<library>.py

☐ README.md

☐ presentation.mp4

☐ contribution.md

Which of the eight has your group not started?

This week. Meet, and hand in the Week 10 minutes. That is the fifth and last.

Then. Finish the library, the tests and the README. Write the paper. Record the presentation.

Week 11. Submit the eight files, as a group.

No more timetabled meetings after this week. What is left is yours to arrange, on your own or together.

The last minutes

Week 10 is the fifth of five

After this week your *I* is whatever the record says

Agree who finishes what

Each of the eight files has a name against it

Record it in the minutes

Anything still unresolved comes to the contact hour. This is the last week it can be fixed cheaply.