What is Testing?

• In Manual testing, a human runs the program and interacts with it to find bugs.

• Automated Testing is the practice of writing code (separate from your actual application code) that invokes the code it tests to help determine if there are any errors.

• It does not prove that code is correct.
Why Testing?

• Testing makes sure your code works properly under a given set of conditions
• Testing allows one to ensure that changes to the code did not break existing functionality
• Good testing requires modular, decoupled code, that is a sign of a good system design
What kind of things can be caught in testing?

- **Syntax errors:** unintentional misuses of the language
- **Logical errors:** created when the algorithm (the way the problem is solved) is not correct.
Unit Testing

• Tests a single “unit” of code.
• A unit could be an entire module, a single class or function, or almost anything in between.

• Consider the following example:
Two functions: is_prime and print_next_prime. Two units

If we want to test print_next_prime, we need first to be sure that is_prime is correct. It is correct?

We write a test for is_prime
import unittest
from primes import is_prime

class PrimesTestCase(unittest.TestCase):
    """Tests for `primes.py`."""

def test_is_five_prime(self):
    """Is five successfully determined to be prime?"""
    self.assertTrue(is_prime(5))

if __name__ == '__main__':
    unittest.main()
Unit tests

- Using the `unittest` Python package, a unit test consists of one or more assertions.
- `self.assertTrue` asserts that the argument passed to it evaluated to True.
- The `unittest.TestCase` class contains a number of assert methods.
- The list could be checked to pick the appropriate methods for your tests.
Unit tests – Fixing Things

• Once we fix the error (for element in range(2, number)), the test runs correctly.

• Now that the error is fixed, does that mean that we should delete the test method? No. unit tests should rarely be deleted as passing tests are the end goal.

• You can write several tests for the same function

```python
def test_is_four_not_prime(self):
    """Is four correctly determined not to be prime?""
    self.assertFalse(is_prime(4), msg='Four is not prime!')
```

```python
def test_is_zero_not_prime(self):
    """Is zero correctly determined not to be prime?""
    self.assertFalse(is_prime(0))
```
Credits

• https://docs.python.org/3/library/unittest.html
• https://www.python-kurs.eu/python3_tests.php
• https://jeffknupp.com/blog/2013/12/09/improve-your-python-understanding-unit-testing/
Sprint 3 – What I expect

• Some planning with:
  • Assignee
  • Estimated Duration

• At the end:
  • Comparison between estimated durations and real duration for all the tasks for all the sprints (to see if the estimations improved)
  • A summary of all the tasks for all the sprints along with the assignee (just to check if the workload inside the group was even).
Testing and Assessment
1) Documentation

• Motivation of the project.
• What is the input and what is the output of the project
• How to install the product (e.g. install Python 3.7, follow the installation tutorial of PM4Py, download from the specified repositories ...)
• How to run the project
2) Unit Tests

- It is important to write some unit tests to check if your code is correct.
- You have the **ALGORITHM** and you have the **SERVICE**.
- The **ALGORITHM** can be tested using unittest (please be as much modular as possible with your code)
- The **SERVICE** can be tested using particular requests (for example using the requests package).
3) Exceptions / Logging

• Bugs are everywhere 😊
• It’s important to have a proper logging mechanism to signal exceptions in the code.
• Exception management:
  • try: except:
  • You can define custom exception types. You can catch custom exception types. You can RAISE custom exceptions.
4) API

• The provision of the web services API are important in order to integrate your product with other products.

• If you want to be professional, look at an API documentation framework (for example **Swagger**)

• Example of API: URI of the service, arguments in the URL, arguments of the POST request, types of the arguments, description of the service.
5) Code Quality

• Internal to Pycharm or through Pylint you have some (configurable) ways to measure the quality of your code.
• If you want to get a really really really really good grade please execute some of these tests.
• When you execute such tests, you have a list of complaints. If you start working the complaints, you get a lower grade.
• PS: please focus on yours code not the distributed engine (that I am aware it gets a low grade 😞 )
6) Internal Code Documentation (as much as possible 😞)

• Helps to describe the method, its input and its output.
• Important for the final Python user.

```python
def check_if_comp_is_completely_unconnected(self, conn1, conn2):
    
    Checks if two connected components are completely unconnected each other

    Parameters
    ------------
    conn1
    | First connected component
    conn2
    | Second connected component

    Returns
    --------
    boolean
    | Boolean value that tells if the two connected components are completely unconnected

    for act1 in conn1:
        for act2 in conn2:
```
7) (Python >= 3.6) Arguments Annotations

• Permits to specify the type of the arguments and of the return type.
• At run-time it changes nothing.
• At development time, it helps you.
• Useful when you return objects from a function and you want to operate with them (Pycharm then tells you which are the methods and variables contained in the object).
7) (Possibly) measure the performance of your product

• Both for algorithms and web services, you can measure the execution time.
• You have several ways to do that, the most cheap is:

```python
import time
aa = time.time()
func()
bb = time.time()
print(bb-aa)
```
8) (If possible) “Professional” deployment of your application 😊

• Maximal freedom is left there:
  • UWSGI
  • Docker (as proposed by some of you at the start of the lesson)

• This is you want to achieve a really really really really good grade (especially MSc).