esade

Advanced Python

Class 4

Pepe Bonet Giner

15th January 2024

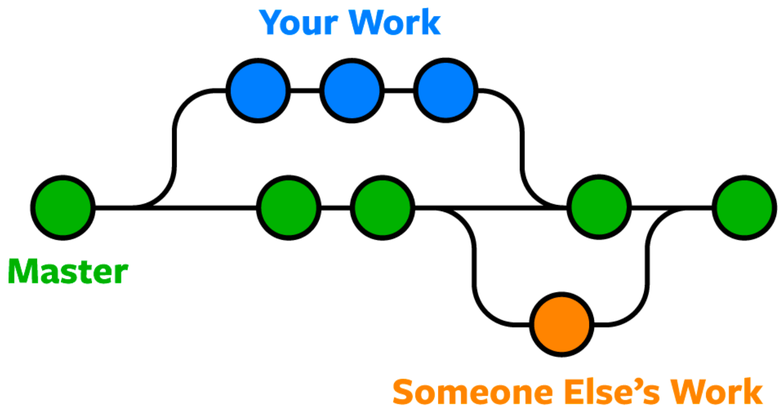
Index Class 4

esade

Topic 1: Recap Class 4

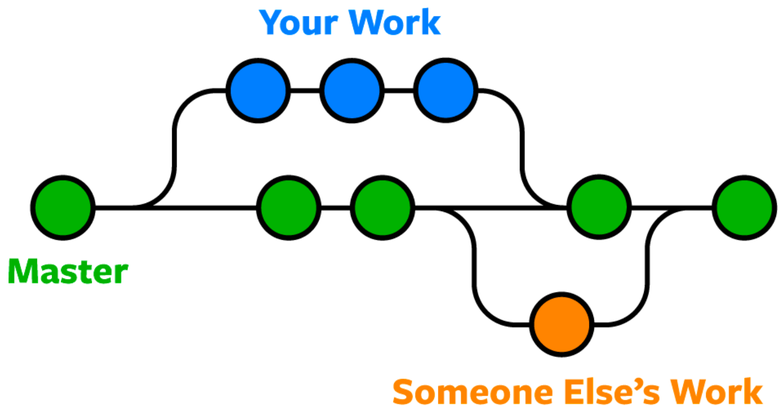
Code Versioning - GitHub

- **Versioning:** Allows developers to track changes to their codebase over time.
- **Collaboration:** It enables multiple developers to work on the same project concurrently, managing code modifications, merging changes, and resolving conflicts efficiently.
- **Backup:** Remote code repository, hosting your project's code and providing a secure backup
- **Continuous Integration and Deployment (CI/CD):** Automated build and testing processes. Ensure code quality, and allows for smooth deployment to production environments.



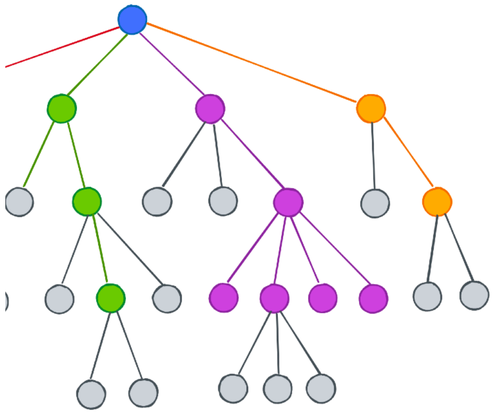
Take home message —> You gotta use GitHub (or similar)

- **If you manage any type of data team or want to in the future**
- **If you want to do any data project**
- **If you want to properly learn how to work on an ML project**
- **If you plan on collaborating with people**
- **If you want to create your own product to share on the internet**



Bringing everything together

CLI Parser (Click)



Debugger (PDB)

Code Working

Code Working

Code Working

Code Working

Add debugger (stop the code)

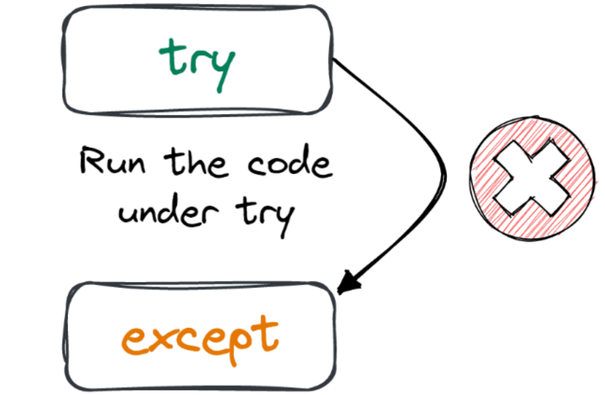
- Inspect the problem
- Solve it

Code Not Working —> Code Working

More Code

More code

Try & Except



When there is an Exception, execute the code under except

Classes

```
Codeium: Explain | CodiumAI: Options | Te
class filters_dataset:

    Codeium: Refactor | Explain | Gener
    def __init__(self, df):
        self.df = df
```

GitHub

Comments on assignments

I am happy with your work

Follow the assignment please

Try to deviate from the code we write in class. Filtering is good but maybe do something else. EDA (figures) - clean up dataset

Bringing everything together

Any questions so far?
Too quick? Too slow?
Do you get the new concepts?
Do you think it can be useful?

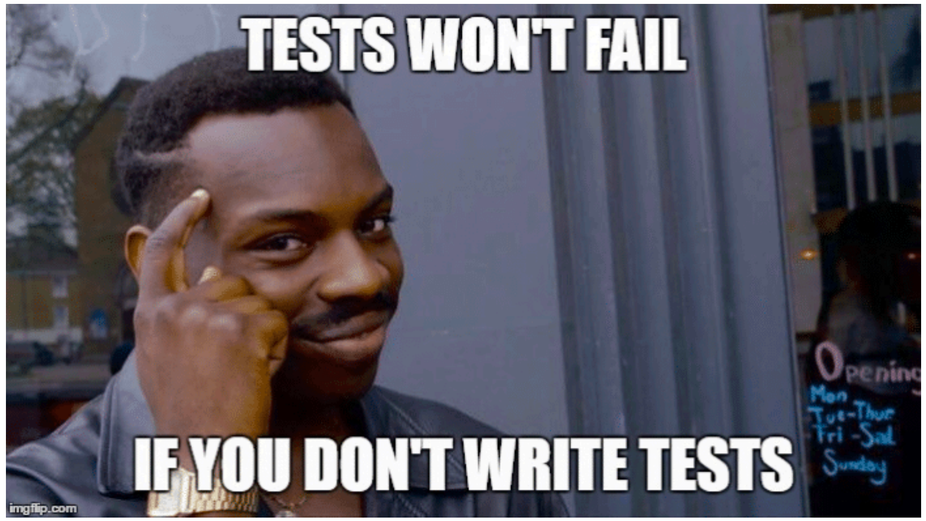
esade

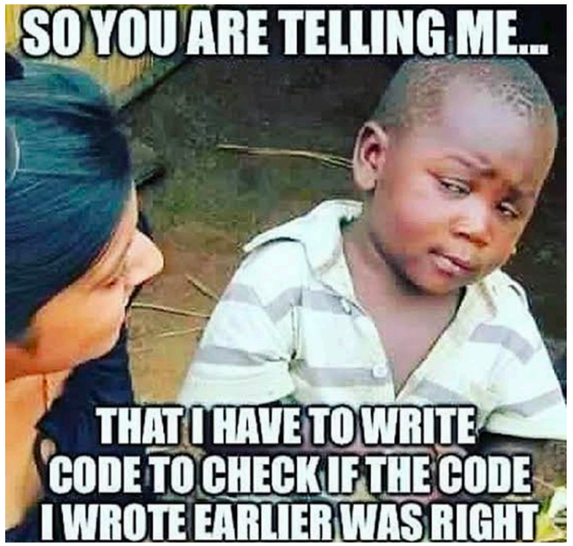
Topic 2: Testing

What does it mean to test your code?

Testing in Python refers to the process of systematically checking and validating that individual units of source code, such as **functions** or methods, work correctly.

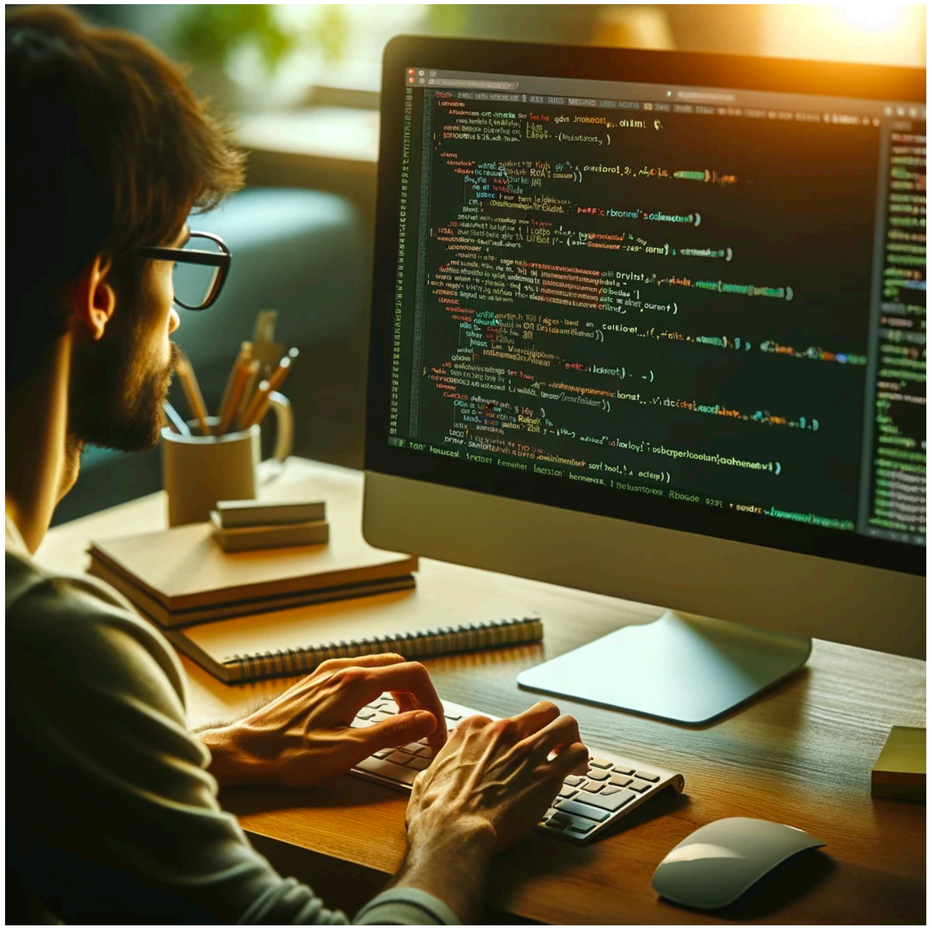
First of all, yeah it kinda sucks to write tests



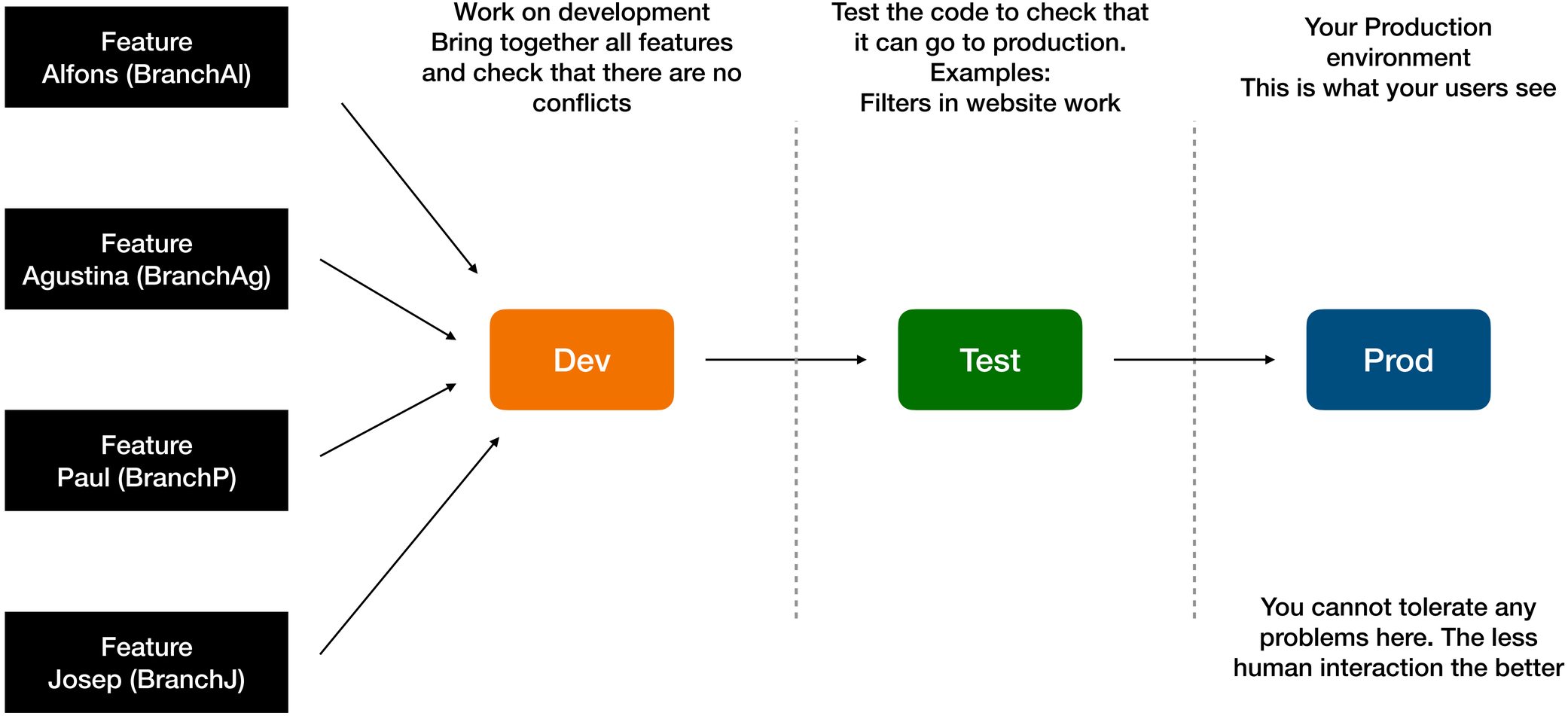


Why is it important to test?

- **Identifies Bugs** Early
- Ensures **Code Quality** and behaves as expected under various conditions.
- Facilitates **Refactoring** (changes) knowing that tests will catch any new errors introduced.
- Improves **Reliability** and **Robustness** of code.
- Facilitates **Integration** & **Deployment**
- Act as Documentation

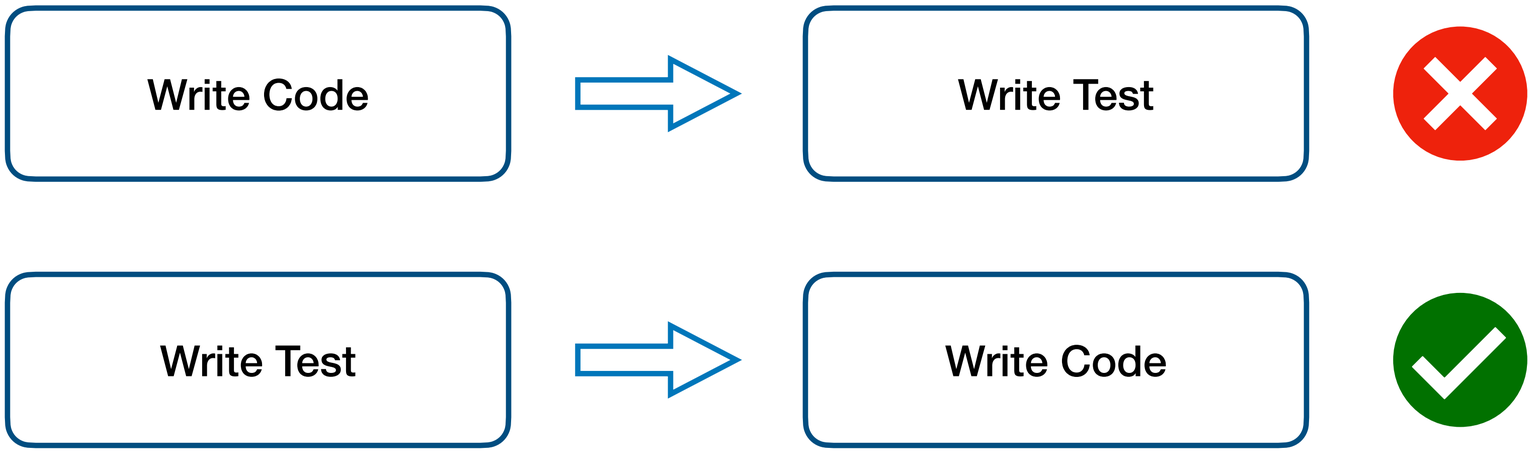


Back to an example product @ Hynts (test has to be there)



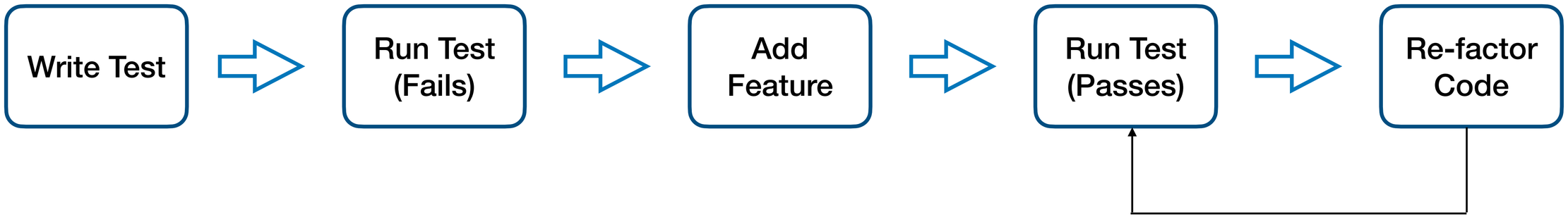
Test Driven Development (TDD) (I struggle to follow this)

- Development Practice



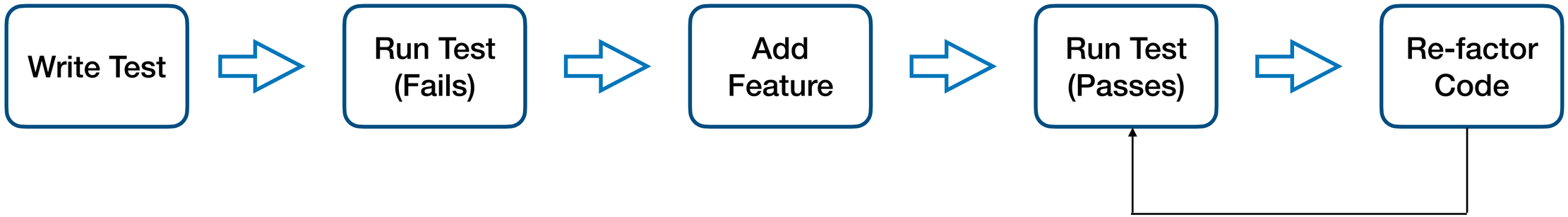
TDD Process

- Development Practice



TDD Process

- Development Practice



- Better understanding of your code
- Better coverage making changes easier (More confidence)
- Reduces Bugs

Unittest and Pytest

- Unittest and Pytest are the main Python packages that we can and will use to test our code.
- We need:

A function to test

```
Codeium: Refactor | Explain | ✕ | CodiumAI: Optic
def sum_numbers(x, y):
    """
    Function to sum two numbers
    """

    return x + y
```

A test for the function

```
def test_sum():
    """
    Function to test sum
    """
    result = ca.sum_numbers(5, 10)

    assert result == 15
```

Follow along

- Let’s build first a simple case scenario where we could use a test.
- First let’s create a branch named class4_tests and work there
- Functions to compute the sum and subtraction of two numbers and test that they work correctly.
- Let’s use both
  - pytest (pip install pytest & the pytest tests (it catches all tests) in the command line)
  - unittest (no install needed + python -m unittest in the command line)

Follow along

- Now, let’s see this in action and build some tests together on how to ensure that the input dataset is correctly passed

Exercise

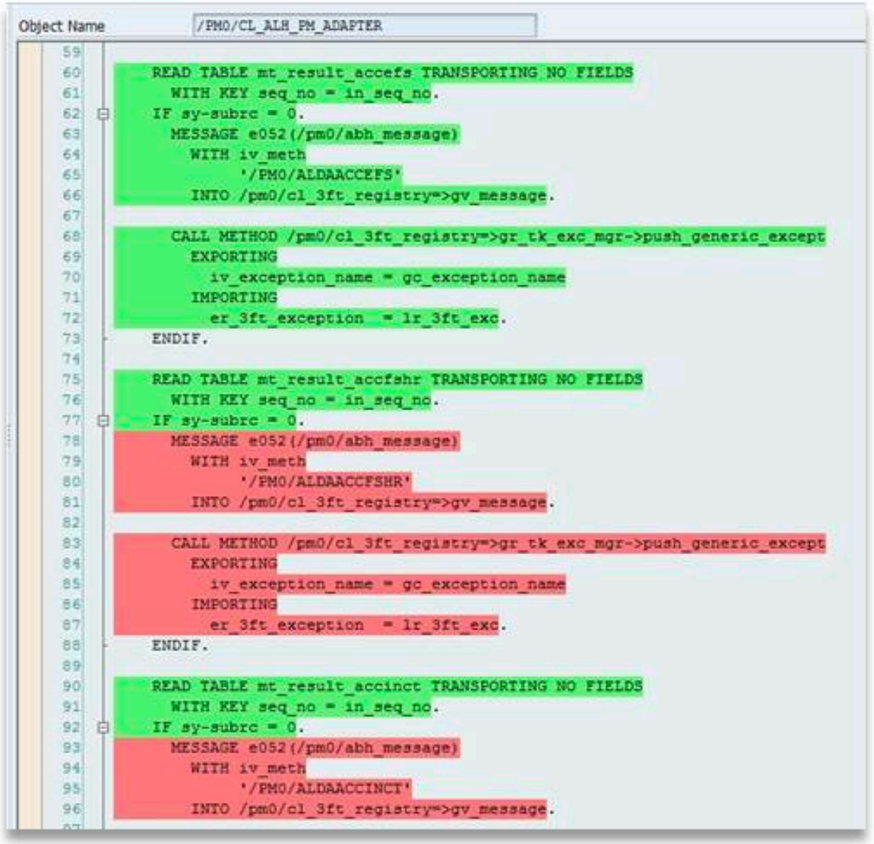
- Build some tests for the filtering steps performed on our dataset.
- If you build a class for filtering functions you can put that class in a different script and call it from the main script
- (Show an example on how you can import code from other files to create scripts with independent functionality)

esade

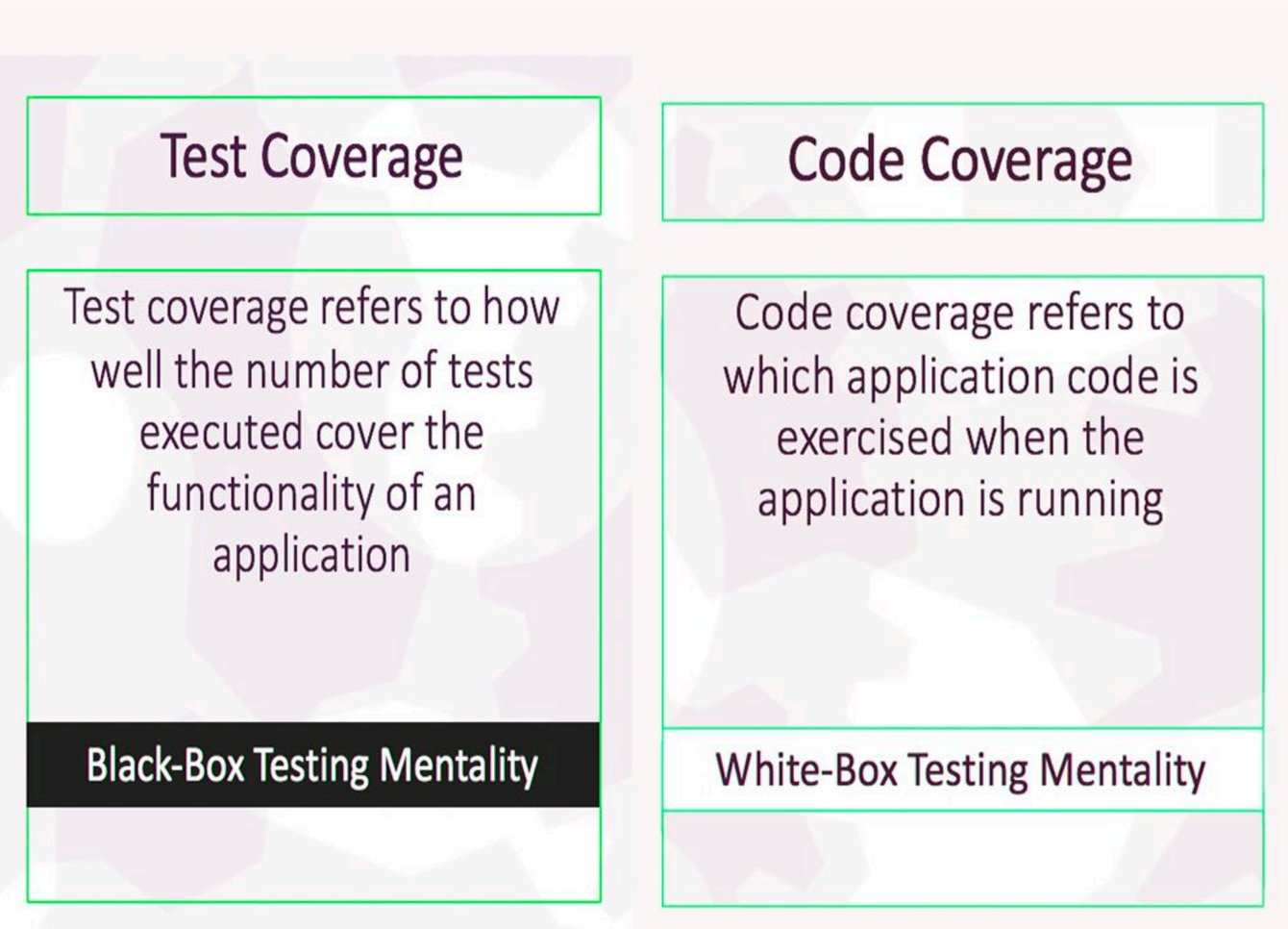
Topic 3: Code Coverage

Code coverage

Code coverage refers to the percentage of your code (number of lines of your code) that is covered by tests (that is assessed).



Code Coverage vs. Test coverage



Installation and Usage

```
pip install coverage==(look for library version)

coverage run -m unittest tests/test_filter.py

coverage report

coverage html
```

Follow along

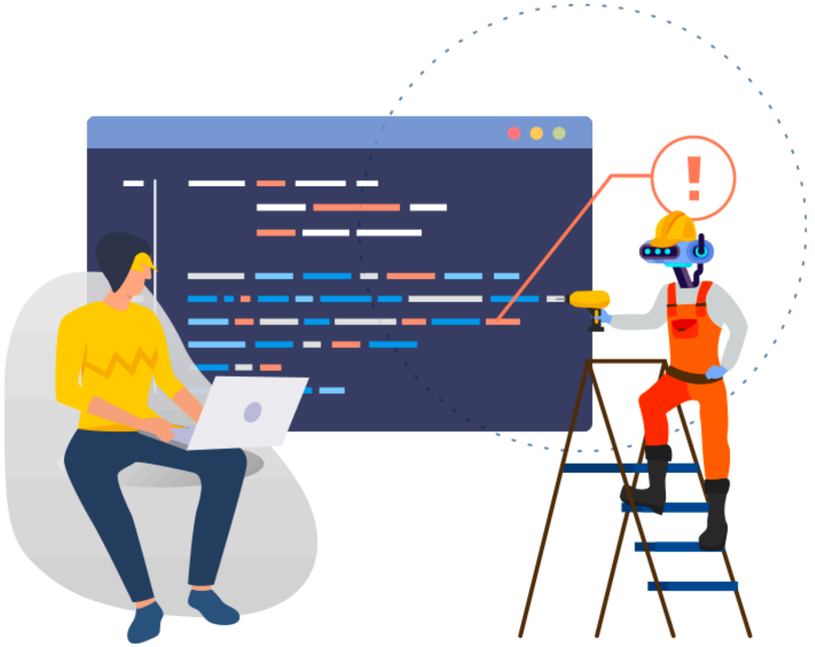
- Let’s check our code coverage and see whether we can improve it

esade

Topic 4: Linting

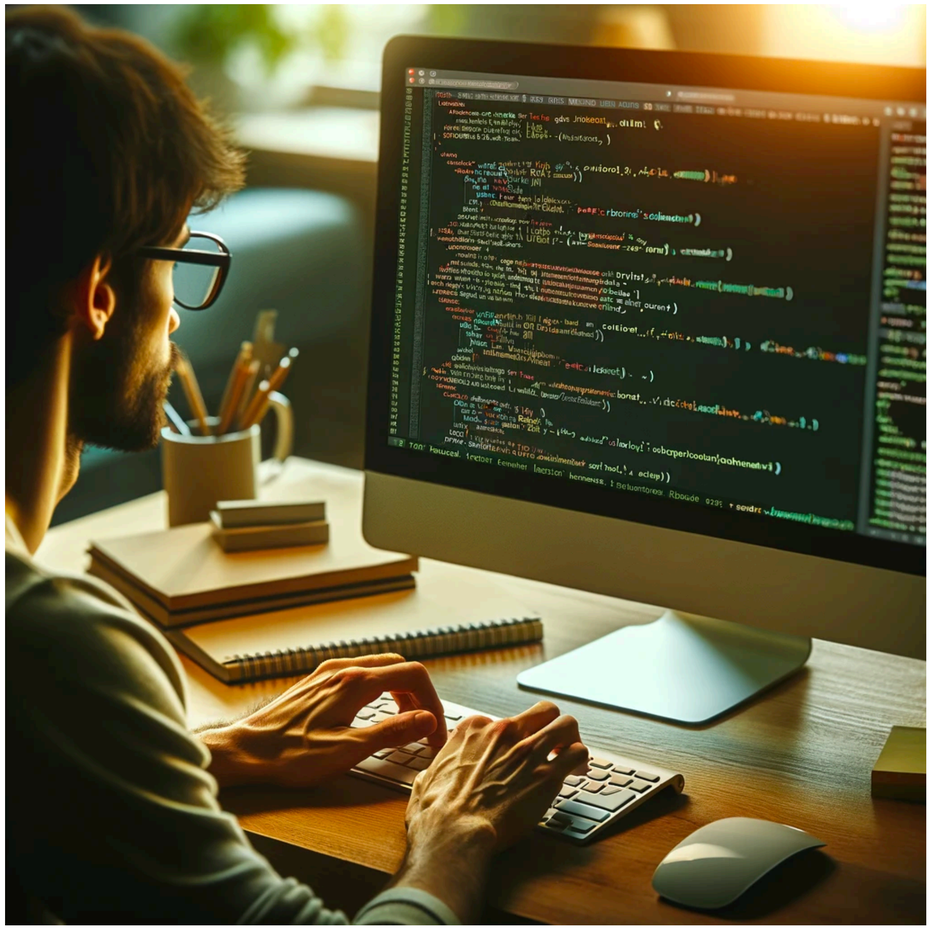
What is linting?

Linting is to run a program that analyzes Python code for various errors and potential problems. It checks for programmatic and stylistic errors, helping developers maintain a clean and consistent codebase that adheres to best practices.



Is linting useful? (Testing is more)

- **Code Quality Improvement**
- **Early Bug Detection**
- **Readability and Maintainability**
- **Facilitates Code Reviews**



Flake8 & Pylint

Both are libraries to lint your code. Their messages are different so it is not bad to use both and implement the changes they suggest

flake8

Command line:

pip install flake8

flake8 folder/to/lint

Command line:

pip install pylint

pylint folder/to/lint

Flake8 & Pylint Outputs

flake8

```
(advancedpython) pepebonetginer@Pepes-MacBook-Pro filmgenre_AdPy % flake8 .
./scripts/calc.py:23:9: W292 no newline at end of file
./scripts/explore_dataset.py:10:1: E302 expected 2 blank lines, found 1
./scripts/explore_dataset.py:52:80: E501 line too long (80 > 79 characters)
./scripts/explore_dataset.py:55:80: E501 line too long (83 > 79 characters)
./scripts/explore_dataset.py:58:80: E501 line too long (82 > 79 characters)
./scripts/explore_dataset.py:66:15: E702 multiple statements on one line (semicolon)
./scripts/explore_dataset.py:66:15: E231 missing whitespace after ';'
./scripts/explore_dataset.py:81:11: W292 no newline at end of file
./tests/test_calc_unittest.py:28:36: W292 no newline at end of file
./tests/test_input_dataset.py:25:5: E303 too many blank lines (2)
```

Pylint
Star your Python code!

```
(advancedpython) pepebonetginer@Pepes-MacBook-Pro filmgenre_AdPy % pylint scripts
************* Module scripts.calc
scripts/calc.py:23:0: C0304: Final newline missing (missing-final-newline)
************* Module scripts.explore_dataset
scripts/explore_dataset.py:81:0: C0304: Final newline missing (missing-final-newline)
scripts/explore_dataset.py:10:0: C0115: Missing class docstring (missing-class-docstring)
scripts/explore_dataset.py:10:0: C0103: Class name "filters_movies" doesn't conform to PascalCase naming style (invalid-name)
scripts/explore_dataset.py:15:4: C0116: Missing function or method docstring (missing-function-docstring)
scripts/explore_dataset.py:10:0: R0903: Too few public methods (1/2) (too-few-public-methods)
scripts/explore_dataset.py:19:0: C0115: Missing class docstring (missing-class-docstring)
scripts/explore_dataset.py:19:0: C0103: Class name "computations_movies" doesn't conform to PascalCase naming style (invalid-name)
scripts/explore_dataset.py:21:4: C0116: Missing function or method docstring (missing-function-docstring)
scripts/explore_dataset.py:66:4: C0415: Import outside toplevel (pdb) (import-outside-toplevel)
scripts/explore_dataset.py:66:15: C0321: More than one statement on a single line (multiple-statements)
scripts/explore_dataset.py:66:15: W1515: Leaving functions creating breakpoints in production code is not recommended (forgotten-debug-statement)
scripts/explore_dataset.py:72:12: W0707: Consider explicitly re-raising using 'raise TypeError(f'The parameter year was not found: {e}') from e' (raise-missing-from)
scripts/explore_dataset.py:81:4: E1120: No value for argument 'dataset' in function call (no-value-for-parameter)
scripts/explore_dataset.py:81:4: E1120: No value for argument 'year' in function call (no-value-for-parameter)
scripts/explore_dataset.py:81:4: E1120: No value for argument 'output' in function call (no-value-for-parameter)

------------------------------------
Your code has been rated at 3.33/10
```

Follow along

- Let’s use both (Pylint & flake8) to understand their outputs and how we can improve our code and make it more clear
- To do linting quicker we can also install black. pip install black + black script (show)

Exercise

- Work on tests, coverage and linting on your second repo.