

Effective Programming Practices for Economists

Software engineering

Reusing test functions

Janoś Gabler and Hans-Martin von Gaudecker

Careful with "or"-style conditions

```
def test_clean_agreement_scale_invalid_data():  
    with pytest.raises(ValueError):  
        _clean_agreement_scale(pd.Series(["-77", "typo"]))
```

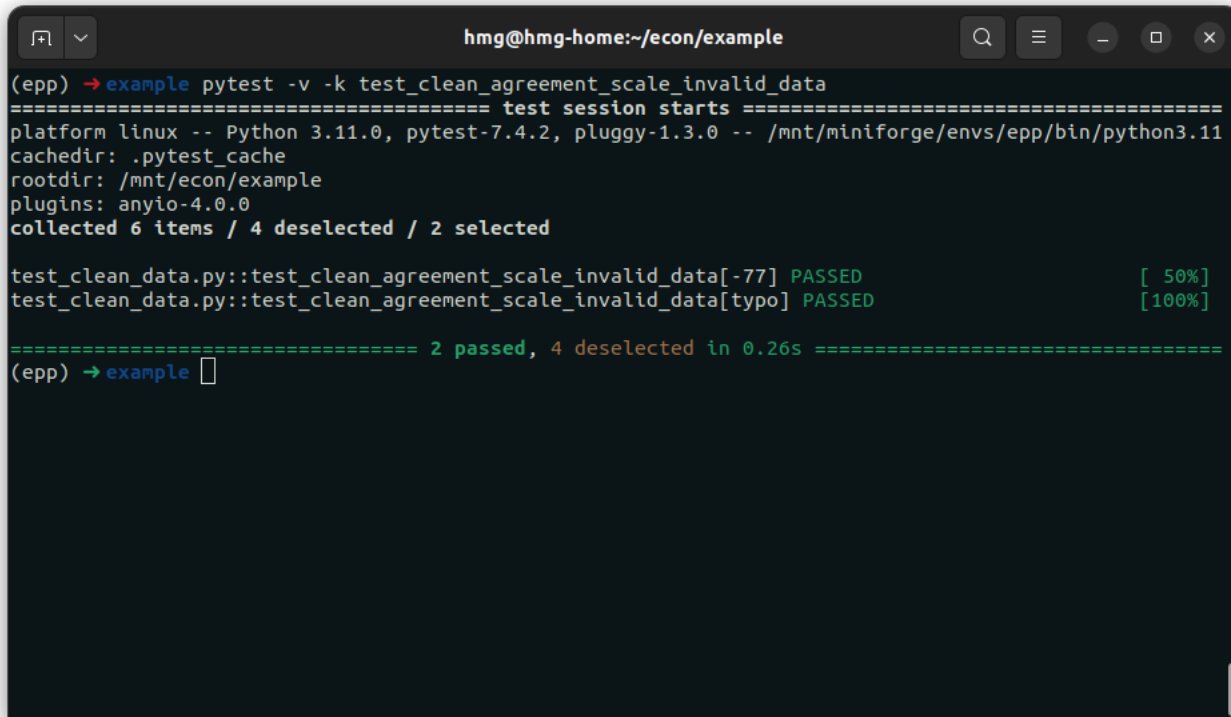
- Could solve by a careful check of message coming with ValueError
- Much clearer: Run once for each element of ``["-77", "typo"]``

Reusing test functions

```
@pytest.mark.parametrize("invalid_input", [-77, "typo"])
def test_clean_agreement_scale_invalid_data(invalid_input):
    with pytest.raises(ValueError):
        _clean_agreement_scale(pd.Series([invalid_input]))
```

- first argument is a string with the test function input
- second argument is an iterable
- test function will be run once for each element of the iterable
- could have more than one argument to test function
 - including expected output
 - see documentation for syntax

One test function, two tests



```
hmg@hmg-home:~/econ/example
(epp) → example pytest -v -k test_clean_agreement_scale_invalid_data
===== test session starts =====
platform linux -- Python 3.11.0, pytest-7.4.2, pluggy-1.3.0 -- /mnt/miniforge/envs/epp/bin/python3.11
cachedir: .pytest_cache
rootdir: /mnt/econ/example
plugins: anyio-4.0.0
collected 6 items / 4 deselected / 2 selected

test_clean_data.py::test_clean_agreement_scale_invalid_data[-77] PASSED [ 50%]
test_clean_data.py::test_clean_agreement_scale_invalid_data[typo] PASSED [100%]

===== 2 passed, 4 deselected in 0.26s =====
(epp) → example
```

Countercheck fails as it is supposed to

