

# **Effective Programming Practices for Economists**

## **Software engineering**

### **Testing code that should raise errors**

Janoś Gabler and Hans-Martin von Gaudecker

# Reminder of the example

```
>>> raw = pd.read_csv("survey.csv")
>>> raw
```

	Q001	Q002	Q003
0	strongly disagree	agree	python
1	strongly agree	strongly agree	Python
2	-77	disagree	R
3	agree	-77	Python
4	-99	-99	Python
5	nan	strongly agree	Python
6	neutral	strongly agree	Python
7	disagree	agree	python
8	strongly disagree	-99	PYTHON
9	-77	-99	Ypython

From the metadata you know

- Q001: I am a coding genius
- Q001: I learned a lot
- Q003: What is your favourite language
  
- -77 not readable
- -99 no reply

# What will happen for invalid data?

```
def _clean_agreement_scale(sr):
    sr = sr.replace(
        {
            "-77": pd.NA,
            "-99": pd.NA
        }
    )
    categories = [
        "strongly disagree",
        "disagree",
        "neutral",
        "agree",
        "strongly agree"
    ]
    dtype = pd.CategoricalDtype(
        categories=categories,
        ordered=True
    )
    return sr.astype(dtype)
```

- What if next year the survey tool changed the representation of missings?
- What if categories were changed?
- What do you actually expect the function to do?

# Tests pin down desired behaviour

```
import pytest

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

- Passing two codes that should not work
- We expect a `ValueError` to be raised
- Test will fail if
  - no error is being raised
  - a different error is being raised

# Run pytest

```
hmg@hmg-home:~/econ/example
(epp) → example pytest
===== test session starts =====
platform linux -- Python 3.11.0, pytest-7.4.2, pluggy-1.3.0
rootdir: /mnt/econ/example
plugins: anyio-4.0.0
collected 5 items

test_clean_data.py ..F.. [100%]

===== FAILURES =====
_____ test_clean_agreement_scale_invalid_data _____

  def test_clean_agreement_scale_invalid_data():
>     with pytest.raises(ValueError):
E       Failed: DID NOT RAISE <class 'ValueError'>

test_clean_data.py:28: Failed
===== short test summary info =====
FAILED test_clean_data.py::test_clean_agreement_scale_invalid_data
- Failed: DID NOT RAISE <class 'ValueError'>
===== 1 failed, 4 passed in 0.30s =====
(epp) → example
```

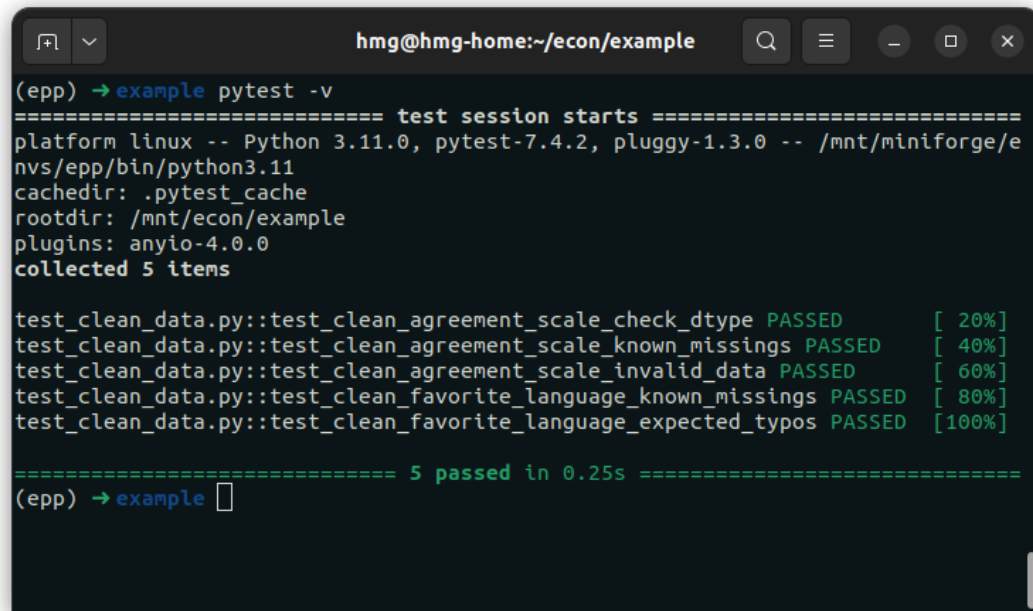
# Tests teach you programmes' behaviour

- This is how I learned that `.astype(pd.CategoricalDtype())` sets values that are not among the categories to missing!
- Small examples are exactly the right level to learn
- Imagine this would have happened in a large project, where you would have noticed only when only 5% of the expected sample size is left in regression tables!
- "Fail early, fail often"

# For the record: Solution

```
def _clean_agreement_scale(sr):  
    known_missings = {"-77", "-99"}  
    categories = ["strongly disagree", "disagree", "neutral", "agree", "strongly agree"]  
    if invalid_values := set(sr.unique()) - set(categories) - known_missings:  
        msg = f"Unexpected values in agreement scale: {invalid_values}"  
        raise ValueError(msg)  
    dtype = pd.CategoricalDtype(categories=categories, ordered=True)  
    return sr.replace({m: pd.NA for m in known_missings}).astype(dtype)
```

# Run pytest, again



```
hmg@hmg-home:~/econ/example
(epp) → example pytest -v
===== test session starts =====
platform linux -- Python 3.11.0, pytest-7.4.2, pluggy-1.3.0 -- /mnt/miniforge/e
nvs/epp/bin/python3.11
cachedir: .pytest_cache
rootdir: /mnt/econ/example
plugins: anyio-4.0.0
collected 5 items

test_clean_data.py::test_clean_agreement_scale_check_dtype PASSED [ 20%]
test_clean_data.py::test_clean_agreement_scale_known_missings PASSED [ 40%]
test_clean_data.py::test_clean_agreement_scale_invalid_data PASSED [ 60%]
test_clean_data.py::test_clean_favorite_language_known_missings PASSED [ 80%]
test_clean_data.py::test_clean_favorite_language_expected_typos PASSED [100%]

===== 5 passed in 0.25s =====
(epp) → example
```