LU04b - Test überspringen

Je nach Situation oder Umgebung wollen wir einzelne Tests nicht durchführen. Mittels `skip` können wir einen Test überspringen.

Siehe auch PyTest - Skipping

Wenn wir im Terminal `pytest` eingeben, werden alle Tests innerhalb unseres Projekt durchgeführt. Es gibt jedoch gute Gründe, weshalb ein Test nicht ausgeführt werden soll. Vielleicht ist die Funktion die getestet werden soll noch gar nicht fertig programmiert. Dann würde der Test ständig als fehlerhaft angezeigt, obwohl wir dies schon im vorhinein wussten. Andere Tests machen vielleicht nur in einer bestimmten Umgebung überhaupt Sinn. Zum Beispiel könnte ein Test nur unter Windows relevant sind und kann unter Linux gar nicht ausgeführt werden.

Decorator

@pytest.mark.skip

Die einfachste Möglichkeit zum Überspringen eines Tests ist der Decorator `@pytest.mark.skip` Damit wird dieser Test in jedem Fall übersprungen. Dieser Dekorator ist sinnvoll, wenn die zu testende Funktion noch fehlerhaft ist und der Test aktuell immer scheitern würde.

```
@pytest.mark.skip
def test_foo():
   ...

@pytest.mark.skip(reason='test currently impossible')
def test_bar():
    ...
```

`skip` hat einen optionalen Parameter `reason='…'` mit dem wir eine Begründung angeben können. Diese Begründung wird dann in der Zusammenfassung von PyTest angezeigt.

@pytest.mark.skipif

Der Dekorator `@pytest.mark.skipif` erlaubt es uns eine Bedingung anzugeben. Ist diese Bedingung erfüllt, so wird der Test übersprungen. Dadurch können wir zum Beispiel einen Test abhängig von der Python-Version ausführen oder nicht.

```
import sys
```

```
@pytest.mark.skipif(sys.version_info < (3, 7), reason='requires python3.7 or higher')
def test_function():
    ...
```

Innerhalb der Funktion

Nebst den Dekorators können wir `pytest.skip` auch innerhalb der Testfunktion aufrufen. Tritt zum Beispiel beim Vorbereiten des Tests ein Problem auf, können wir die weitere Ausführung des Tests überspringen. Auch bei relativ komplexen Bedingungen ist diese Variante besser geeignet als der Dekorator mit `pytest.mark.skipif`.

```
def test_setup(monkeypatch, capsys, test_values):
    if isinstance(test_values, dict) and \
            isinstance(test_values['energy_price'], float) and \
            isinstance(test_values['client1'], str) and \
            isinstance(test_values['client2'], str) and \
            isinstance(test_values['client3'], str):
        print('\tSetup erfolgreich')
    else:
        pytest.skip('\tDie Datei "testdata.json" fehlt oder ist fehlerhaft. Kontaktieren Sie Ihre Lehrperson.')


@pytest.fixture
def test_values():
    try:
        with open('./testdata.json') as json_file:
            data = json.load(json_file)
            data['energy_price'] = float(data['energy_price'])
            return data
    except FileNotFoundError:
        return 'Die Datei "testdata.json" fehlt. Kontaktieren Sie Ihre Lehrperson'
```

M450-LU04