

# LU09.A00: PyTests finden



Erstelle ein JSON-Array mit allen Unittests in einem Python-Projekt.

## Aufgabe

Das Skript durchsucht ein Projekt nach Dateien, die Unitests enthalten. Die Dateinamen müssen `test_` oder `_test` enthalten. Zum Beispiel:

- `test_main.py`
- `main_test.py`

Innerhalb dieser Dateien sucht das Skript nach Funktionen, deren Bezeichner mit `test_` beginnt. Die Bezeichner dieser Funktionen werden in eine Liste geschrieben. Schlussendlich wird diese Liste sortiert und als JSON-Array ausgegeben.

## Beispiel

```
[  
    "test_figure_init_empty",  
    "test_figure_add_shape",  
    "test_figure_add_shape_multiple",  
    "test_figure_take_shape",  
    "test_figure_remove_shape_by_title",  
    "test_figure_remove_shape_by_index",  
    "test_figure_total_area_scale1",  
    "test_figure_total_area_scale3",  
    "test_figureshape_init_empty",  
    "test_figureshape_set_figure",  
    "test_figureshape_set_shape",  
    "test_main"  
]
```

## Wieso Python und Bash?

Grundsätzlich könnte man die Aufgabe ausschliesslich mit Python lösen oder als reines Bash-Skript schreiben. Jede Sprache hat jedoch ihre Vorteile, die wir hier ausnutzen wollen:

- Dateien durchsuchen: Hier spielt Bash seine Stärke aus
- Arbeiten mit Collections: Dies lässt sich leichter in Python umsetzen.

# Umsetzung

## Vorbereitung

1. Klonen Sie das Repository im WSL in einen neuen Ordner.
2. Erstelle das virtual Environment.
3. Führe das Bash-Skript `setup.sh` aus. Dieses erstellt einen Ordner und Dateien für die Tests.

## Teilschritte

In den folgenden Aufgaben werden Sie zunächst einzelne Teilschritte umsetzen und testen. Zum Schluss schreiben Sie die `main`-Funktion um alle Teilfunktionen zu verbinden.

## Hinweise

Siehe [https://wiki.bzz.ch/modul/m122/learningunits/lu08/linux\\_python#virtual\\_environment](https://wiki.bzz.ch/modul/m122/learningunits/lu08/linux_python#virtual_environment)



Das Python-Skript kann nicht unter Windows ausgeführt werden, da wir Bash-Befehle nutzen.

Um auf das Windows-Laufwerk C zuzugreifen, verwendest du im WSL den Pfad `/mnt/c`.

==== Aufruf des Skripts ====  
- Öffne das Windows Subsystem for Linux (WSL). - Wechsel in den Ordner, in dem Ihr Projekt gespeichert ist. Zum Beispiel: `cd „/mnt/c/BZZ/Python/m122-lu08-a02-listtest-ghwalin“` - Starte das Python Skript mit `python3 main.py ORDNER_PFAD`. Ersetze `ORDNER_PFAD` durch den Pfad zum gewünschten Projekt.  
==== Unitests aufrufen ====  
Um einen bestimmten Unitest durchzuführen, wird dieser im Terminal mit dem Befehl `pytest` aufgerufen. `<code bash> pytest MODULE::FUNCTION </code>` \* Ersetze `MODULE` durch den Dateinamen mit den Unitests, z.B. `main_test.py` \* Ersetze `FUNCTION` durch den Namen der Test-Funktion, z.B. `test_find_test_modules` \* Gibst du nur den Modulnamen an, werden alle Tests in dieser Datei durchgeführt. `<code bash> pytest main_test.py::test_find_test_modules pytest main_test.py </code>`

From:  
<https://wiki.bzz.ch/> - BZZ - Modulwiki

Permanent link:  
[https://wiki.bzz.ch/modul/m122/learningunits/lu09/aufgaben/find\\_tests?rev=1736782436](https://wiki.bzz.ch/modul/m122/learningunits/lu09/aufgaben/find_tests?rev=1736782436)

Last update: **2025/01/13 16:33**

