***Pytest Unit Testing: Execution Techniques***

In case we have the following directory structure:

**SE Lab**

**├── Source Code**

**│ └── calculator.py**

**└── Test**

**└── test\_calculator.py**

***Case 1: Running Using Setting Python Path***

To run the test directly, the can set the PYTHONPATH environment variable to include the directory containing the source code.

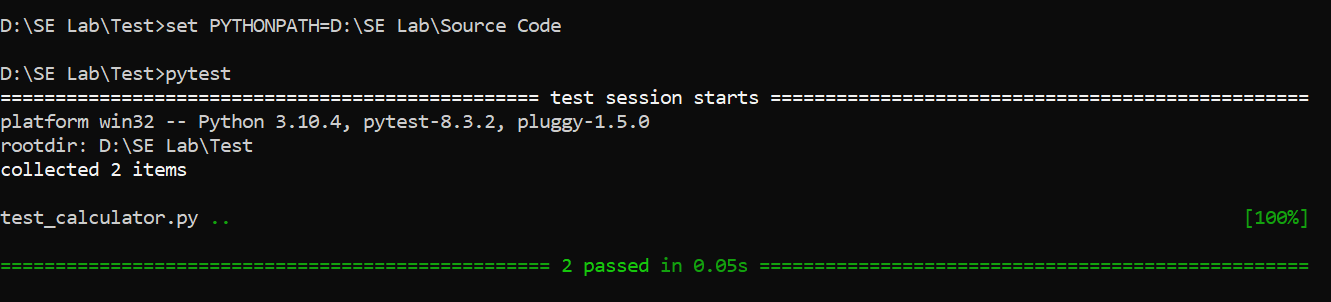
In this example, we would use the following command:

**Set PYTHONPATH=path\_to\_source\_code\_directory**

In this case it is:

**Set PYTHONPATH=D:\SE Lab\Source Code**

This allows Python to easily locate the calculator.py script, enabling the to run the test code without issues.



***Fig 1: Running Pytest in Command Prompt***

***Case 2: Running Tests by Setting the Absolute File Path***

Another approach is to run the test by specifying the absolute path to the source code file directly. This ensures that the calculator.py script is correctly referenced when executing the tests.

import sys

import os

import pytest

# Add the parent directory of 'source' to

sys.path sys.path.insert(0, os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), '..')))

# Importing functions from module from the directory

from Source Code.calculator import add, subtract

After this the path will easy be accessed and the script will be accessed and the test codes will be easily run

In case we have the following directory structure:

**SE Lab**

**├── Source Code**

**│ ├── \_\_init\_\_.py**

**│ └── calculator.py**

**└── Test**

**├── \_\_init\_\_.py**

**└── test\_calculator.py**

***Case 1: Running Using Pytest***

Since \_\_init\_\_.py is already set up, Python will treat the directory as a module and locate the script accordingly. Simply navigate to the base directory, SE Lab, and execute the pytest command. The program will run smoothly without any issues.

**D:\SE Lab> pytest**