Unit Tests

Write three functions

In your lessons folder, create a *folder* called unit_tests.
In unit_tests, create a file called list_fns.py
Write three functions:

- get_first: takes a list[str] as input and returns first element
- remove_first: takes a list[str] as input and removes first element (doesn't return anything)
- get_and_remove_first: takes a list[str] as input and returns + removes first element

Test them out to see if they work! :)

Unit Testing

- Writing functions to test other functions
- Using pytest (other frameworks exist)

Syntax: Writing a unit test

Test file names: end with _test.py
Test function names: begin with test_

```
def test_name() -> None:
    assert <boolean expression>
```

Syntax: Writing a unit test

Test file names: end with _test.py
Test function names: begin with test_

```
def test_name() -> None:
    # Other code can go here!
    assert <boolean expression>
```

Two basic function behaviors you may want to test

- What it returns (Testing For Desired Output)
- How it modifies the input (Testing For Desired Behavior*)

Testing For Desired Output

- Checking that your function returns the correct output given a specific input

Example in VSCode…

Testing For Desired Behavior

- Checking that your function does what you want it to do rather than just checking what it returns.
- This can be useful for functions that *mutate* their input.

Example in VSCode…

Types of Tests

Use case: testing properties for how we expect our program to be used
Edge case: testing instances outside “typical” usage (e.g. empty inputs, incorrect inputs, etc.)

Both can test desired output or behavior!

Syntax: Running a unit test in the terminal

Command line: python -m pytest path/to/testfile.py