Unit Testing

testing a fx to ensure it works.

example:

```
def testiseven()
assert is even(4) == True
```

Testing with PYTEST

-> a python library for automated test

-> detailed feedback in a systematized way.

↳ important to understand why it failed

ASSERT:

- verifies the boolean condition is True.
  if it's FALSE you get an error.
- helps detect error quickly in the dvp of long code.

ex ① `x = 1 + 1`
`assert x == 2`
`print(x)`
w/ you get an error ⇒ its not true

② `n = 5`
`assert n ≥ 0, when we want t`
`print(n)`

So you use "assert" to see if you get errors when you run -> you know code is incorrect
So you assert a condition contrary to your final desired outcome

Conditions:

pre-conditions:
sqrt(x) requires $x \geq 0$

post condition
x^2 will have to be > 0

Testing fxs:

verify fx work under all expected conditions.
↳ ensures modular design.

Automated Unit Tests:

-> repeatedly run test automatically
-> saves time

ex:

```
def fx....
  if result == 10:
      print("Test 1: PASS")
  else:
      print("Test 1: Fail")
```

with ASSERT: instead assert result==10
so you use assert pass -> value you want
assert fail -> value x want