Unit Testing

https://github.com/chezmarcbrown/classdemo-2022-12-06-testing.git

```
import math

def is_prime(n):
    if n < 2:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True
```

Test in Interpreter -

```
from prime import is_prime
is_prime(1)
is_prime(2)
is_prime(8)
is_prime(11)
```

Better:

```
    for i in range(2, int(math.sqrt(n))):
```

Test again in Interpreter -

```
from prime import is_prime
is_prime(1)
is_prime(2)
is_prime(8)
is_prime(11)
```

Part A: Implement a test function (tests0.py - test_prime)

Part B: Implement a slew of tests (tests0.sh)

```
python3 -c "from tests0 import test_prime; test_prime(1, False)"
python3 -c "from tests0 import test_prime; test_prime(2, True)"
```

```
python3 -c "from tests0 import test_prime; test_prime(8, False)"
python3 -c "from tests0 import test_prime; test_prime(11, True)"
```

Find the bug…

Framework for testing… unittest:ptest1.py

Go back to django…

airline/flights/models.py

Added is_valid_flight(self)...

Want to test all sorts of other relationships as well…

Look at: **airline/flights/tests.py**

Run the tests: **python manage.py test**

Need to fix `is_valid_flight`

Want to now test webpage

(2nd half of the tests)

Change context variable

Frameworks for testing webbrowser; selenium

Counter.html… now add:

```
<script>

    document.addEventListener('DOMContentLoaded', () =>  {

        let counter = 0;

        document.querySelector('#increase').onclick = () => {
            counter++;
            document.querySelector('h1').innerHTML = counter;
        };

        document.querySelector('#decrease').onclick = () => {
            counter--;
            document.querySelector('h1').innerHTML = counter;
        };
    });
</script>

<button id="increase">+</button>
<button id="decrease">-</button>
```

Simulate a user driving a web:

```
>>> from tests import *
>>> uri = file_uri("counter.html")
>>> driver.get(uri)
>>> driver.page_source(uri)
```

Simulate a user driving a web:

```
increase = driver.find_element("id", "increase")
increase.click()
>>> for i in range(100):
...     increase.click()
```

Check what should be on the page:

```
driver.find_element("tag name", "h1")
driver.find_element("tag name", "h1").text
 self.assertEqual(driver.find_element(By.TAG_NAME, "h1").text, "3")
```