Unit Testing 2 cheatsheet

Dependency Injection

Pre Dependency Injection

```
def get_countries():
    headers = {'Content-Type': 'application/json'}
    api_url = "https://restcountries.com/v2/all"

    response = requests.get(api_url, headers=headers)

    if response.status_code == 200:
        return response.json()
    else:
        return None

def get_country_code(key):
    countries = get_countries() # Dependency
    return countries[key]
```

Post Dependency Injection

```
def get_countries():
    headers = {'Content-Type': 'application/json'}
    api_url = "https://restcountries.com/v2/all"

    response = requests.get(api_url, headers=headers)

    if response.status_code == 200:
        return response.json()
    else:
        return None

 # Inject get_countries dependency
def get_country_code(key, get_countries):
    countries = get_countries() # Execute dependency
    return countries[key]

print(get_country_code("GBR", get_countries))
```

- When we call `get_country_code` in our application, we inject the real `get_countries` function
- When we call `get_country_code` in our test, we inject a fake (*mock*) `get_countries` function

Writing Test for Dependency Injection

```
def test_find_country_capital():
    # Arrange
    expected = {"alpha3Code": "GBR", "capital": "London"}

    def mock_get_countries():
        return [
            {"alpha3Code": "USA", "capital": "Washington DC"},
            {"alpha3Code": "GBR", "capital": "London"},
            {"alpha3Code": "TKM", "capital": "Ashgabat"}
        ]

    # Act
    actual = get_country_code('GBR', mock_get_countries)

    # Assert
    assert actual == expected

test_find_country_capital()
```

Unit Testing Terms and Definitions

- `Mock`: A piece of *fake* code standing in to replace some *real* code.
- `Stub`: Dummy data serving to replace real data usually returned from an external source.
- `Dependency`: A piece of code relied upon by another piece of code.
- `Dependency Injection`: A Software Development paradigm in which dependencies are passed as inputs into the function/class that invokes them.