# Object Oriented Programming with Python

Gramsci Hermozo

Session 13

# Content

- Unit test

# Unit test

### unittest

`unittest` is python library that help us to create unit tests for our application. This package cames by default with python instalation.

## unittest

### How to name unit test in python?

We need to follow some conventions to name the tests.

```
*test.py
*_test.py
test*.py
test_*.py
*test*.py
```

# unittest

### How to write the unit test?

```
import unittest

class TestObject(unittest.TestCase)
  pass
```

# unittest

## How to name methods for unit test?

I use to split the function naming in two:

```python
# basic tests
#test_<function_name>
def test_add(self):
  pass


# negative tests
# test_<function_name>_<expecter_results>
def test_divide_raise_value_error(self):
  pass
```

## unittest

### Asserts

assert* functions are used to validate that the value returned is something that we expect. Below some commong functions.

```python
self.assertEqual(<result>, <expected_result>)
self.assertTrue(<result>)
self.assertFalse(<result>)
self.assertRaises(<exception>, <function>, <params>, )
self.assertIn(<value>, <list/dict>)
```

# unittest

## Setup and Teardown

Sometimes we want to prepare a context for each test to be run
under. The setUp method is run prior to each test in the class and
tearDown is run at the end of every test. This methods are
optional.

```
import unittest

class SomeTest(unittest.TestCase):
  def serUp(self):
    self.mock_data = [1, 2, 3, 4, 5]
  def tearDown(self):
    self.mock_data = []
  def test(self):
    self.assertEqual(len(self.mock_data), 5)
```

# unittest

### mock -> patch

The `patch` decorator/context manager makes it easy to mock
classes or objects in a module under test
`from unittest.mock import patch`

```python
def test(mock_class1, mock_class2):
  pass
```
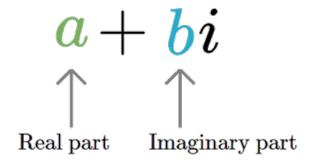
# Lets practice

## Complex numbers

Implement your object complex numbers and add the following function:

- Adding (numComplex1 + numComplex2)
- Multiplying (numComplex1 * numComplex2)
- str/print

# What is a complex number

Is a combination of a **Real Number** and an **Imaginary Number**



$$a + bi$$

Real part     Imaginary part

# operation with complex number

Adding:
$$(a + bi) + (c + di) = (a + c) + (b + d)i$$

Multiplying:
$$(a + bi)(c + di) = ac + adi + bci + bdi2$$

**Note:** i2 = -1