IMPERIAL

BIOE50010 – Programming 2

Computer Lab 9

Binghuan Li Department of Chemical Engineering
Maria Portela Department of Bioengineering
Wenhao Ding Department of Bioengineering

2 December, 2024

Progress Check

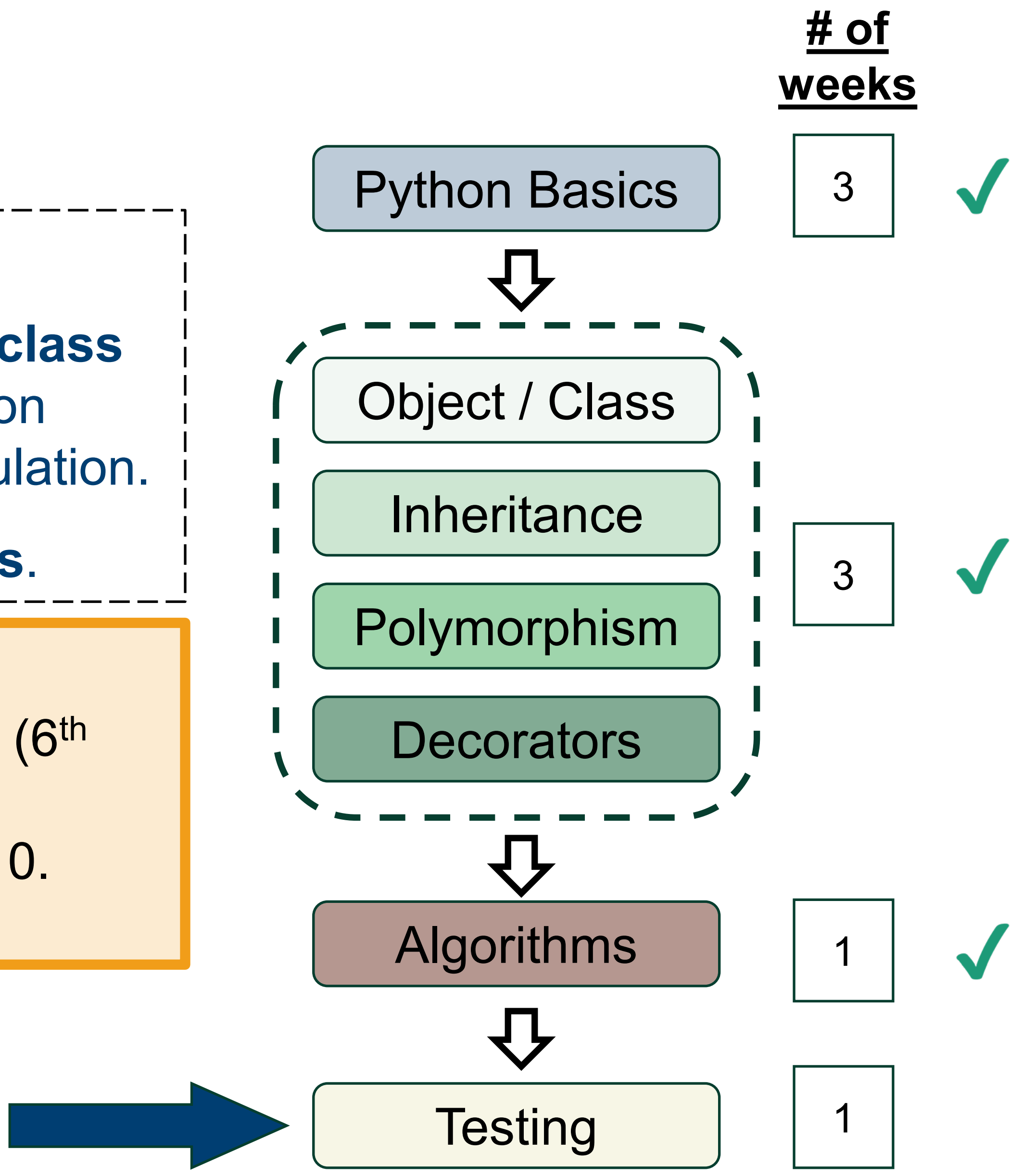
Checklist: you should have mastered…

- How to use implement a **static method**, a **class method**, define a getter and a setter function (**property**) to facilitate better class encapsulation.
- How to defined and use **wrapper functions**.

N.B.

- The assignment will be released on **Friday** (6th Dec, 2024).
- There will be no additional tasks for week 10. Labs will be running in a **Q&A mode**.

Week 9:
we are here



Your task today

Generate the test examples, and creats a test suite using module **`unittest`**, perform comprehensive tests to two functions **`eval_win()`** and **`board_full()`** in the Tic Tac Toe game.

To start…

- Read and study the example Python scripts from your Friday lecture.
- The functions subject to test are given out in TicTacToe.py on Blackboard. To start, import them to your script.
- Consult the summaries of the unit test methods (given out the subsequent pages), when necessary.

Unit Test

To define the test cases using **`unittest`**

- Each test case should be defined as a method, with its name starting with the keyword '**test**'.
- A series of assertion methods have been defined in **`unittest.TestCase`** class – hence use inheritance to access to these methods.

Example from `test_point_pp.py`

```python
import unittest
import point_pp as point

class TestPointPP(unittest.TestCase):

    def test_add(self):
        result = point.add([10, 2],[1, 7])
        self.assertEqual(result, [11, 9])
```

Driver (test runner)

```python
if __name__ == "__main__":
    unittest.main()
```

- You can define multiple test cases within one test class.
- All test cases will run automatically `unittest.main()`

Unit Test Methods

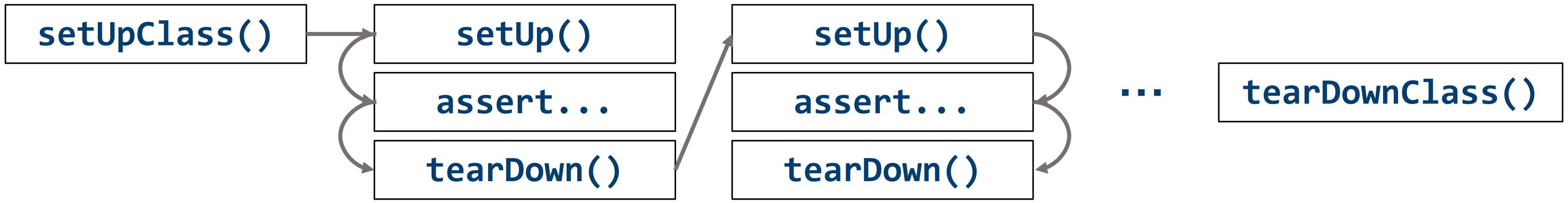
- **Test assertion methods**

unittest method	Checks that…	unittest method	Checks that…
`assertEqual(a,b)`	`a == b`	`assertIsNone(x)`	`x is None`
`assertNotEqual(a,b)`	`a != b`	`assertIsNotNone(x)`	`x is not None`
`assertTrue(x)`	`bool(x) is True`	`assertIn(a, b)`	`a in b`
`assertFalse(x)`	`bool(x) is False`	`assertNotIn(a,b)`	`a not in b`
`assertIs(a,b)`	`a is b`	`assertIsInstance(a,b)`	`isinstance(a, b)`
`assertIs(a,b)`	`a is b`	`assertNotIsInstance(a,b)`	`not isinstance(a, b)`
`assertIsNot(a,b)`	`a is not b`		

Unit Test Methods

- **Test fixature methods**

Method	Description
`setUp()`	The method is called automatically before running *each* test method in a test case class.
`tearDown()`	The method is called automatically after running *each* test method in a test case class.
`setUpClass()`	The method is called automatically before running the tests in a test case class.
`tearDownClass()`	The method is called automatically after running the tests in a test case class.


setUpClass()
setUp()
assert...
tearDown()
setUp()
assert...
tearDown()
...
tearDownClass()