IMPERIAL

BIOE50010 – Programming 2

Computer Lab 6

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

9 November, 2024

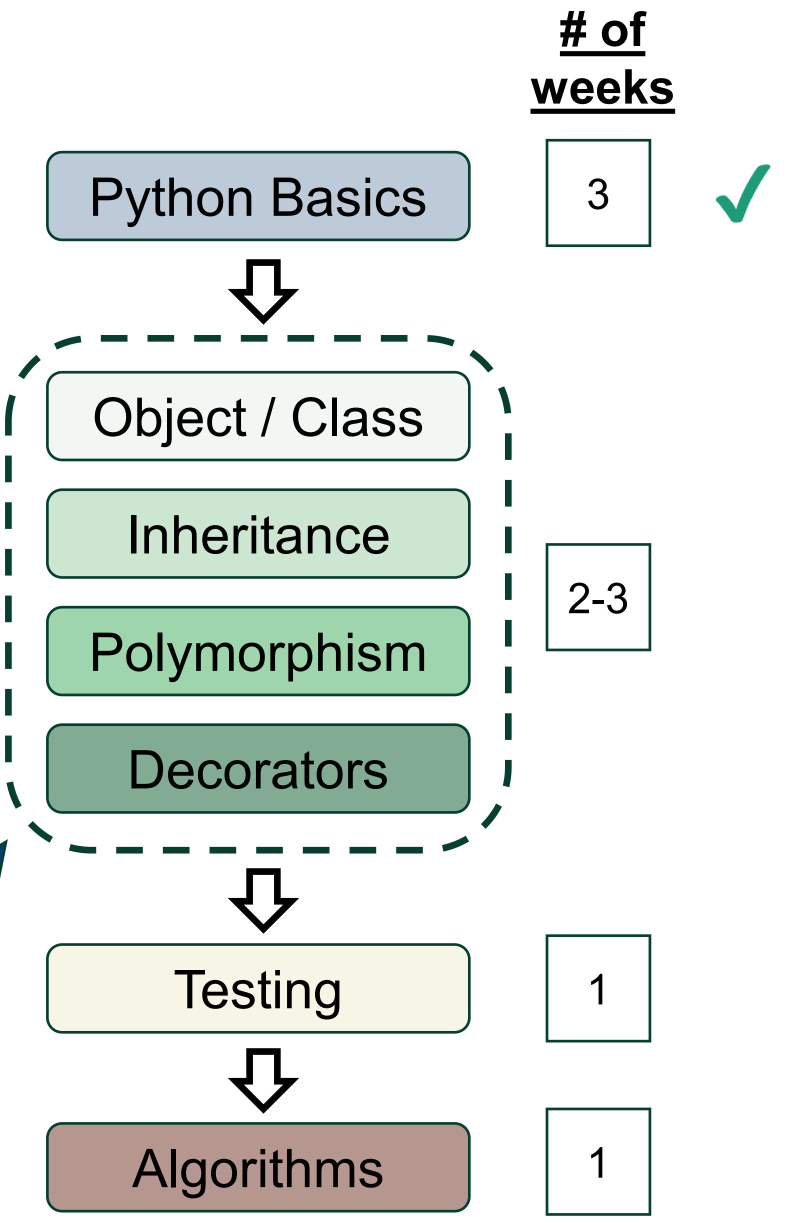
Progress Check

Checklist: you should have mastered…

- **Four pillars of OOP:** abstraction, encapsulation, inheritance, polymorphism.
- **Syntax/concepts of coding inheritance in Python:** super class, sub-class, `super()` function
- **Inheritance can be in many forms:** single, multiple, multi-level.

Questions outside the classroom? ed discussion

Week 6:
we are here



Introduction to Exception Handling

- An **exception** is an error that happens during the execution of a program.
  - Exception: usually from the programme-level, *e.g.*, bugs. See here for a summary
  - Error: usually from the system-level, *e.g.*, not enough memory
- In Python, exceptions can be handled using the **`try…except…`** clause

Example: use of `try…except…` clause

```
while True:
    try:
        x = int(input("Please enter a number (1-9): "))
        break
    except ValueError:
        print("That was not valid number. Try again...")
```

`ValueError` raises due to the failure of typecasting, *e.g.*, typecast a string to an integer

- A more complex exception handling syntax is **`try… except… finally…`**

Your task today

- Bioinformatics (DNA database) with **object-oriented programming**.
    - File I/O: load and read the contents from a .fna file into Python
    - Manipulating the DNA data: concatenation, indexing, counting…

To start…

- Recall the **string / list methods** and **file I/O methods** you have used (Lab 2 slides)
- Recall the class **special methods** and operator overloading you have used (Lab 4 slides)
- Read all information and the **sample output** provided in the lab carefully
- Try to integrate **exception handling** into your code: *e.g.*, “open a file and read in the data; if your attempt fails, return an empty data structure.”

Questions?

That's it for now.

You can now proceed to the Lab 6 exercises.

Summary of Common Exceptions

Exception	Description
`AttributeError`	Accessing an undefined attribute in a **class**.
`ImportError`	**Module** import fails.
`IndexError`	Accessing an out-of-range index in a **list** or **tuple**.
`KeyError`	Accessing a non-existent **dictionary** key.
`NameError`	Using a **variable** that hasn't been defined.
`TypeError`	Performing an operation on an inappropriate **data type**.
`ValueError`	Passing a valid type but **invalid value**.
`ZeroDivisionError`	Dividing by zero.
`SyntaxError`	Code contains a **syntax** error.
`RuntimeError`	**Generic** error for code execution issues.

Advanced: ‘Is a’ or ‘Has a’?

🟥 new terminology!

- Consider the following associations between two objects:

“Dog **is a** breed of mammal” ⇨ Compatible with **inheritance**

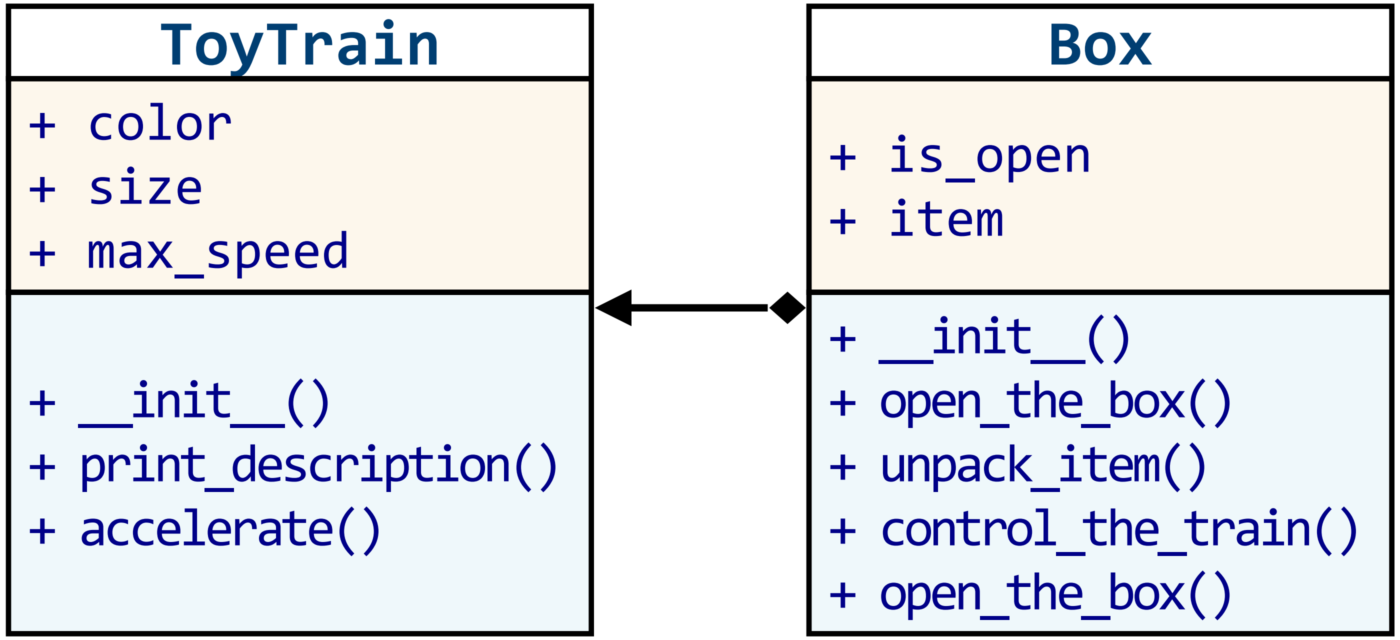
“The car **has an** engine.” ⇨ Not compatible with inheritance!

- The second type of association (‘***has-a***’ relation) is more appropriate where one object is a component or part of another, rather than being a type of that object.
- This relation is known as the **composition**.

Advanced: Composition

🟥 new terminology!

- **Composition** is a strong form of association where a class contains objects of another class as part of its internal structure.
- The following code example…



- **`Box.item`** is an attribute holding the **`ToyTrain`** object.
- Therefore, the following expressions are functionally equivalent:

```
ToyTrain.accelerate()
Box.item.accelerate()
```

See `Box_example_composition.py` on Bb