UNIVERSITÉ CÔTE D'AZUR

Agent-Based Modeling (Master SIED)

Andrea G. B. Tettamanzi
I3S Laboratory – SPARKS Team
`andrea.tettamanzi@univ-cotedazur.fr`

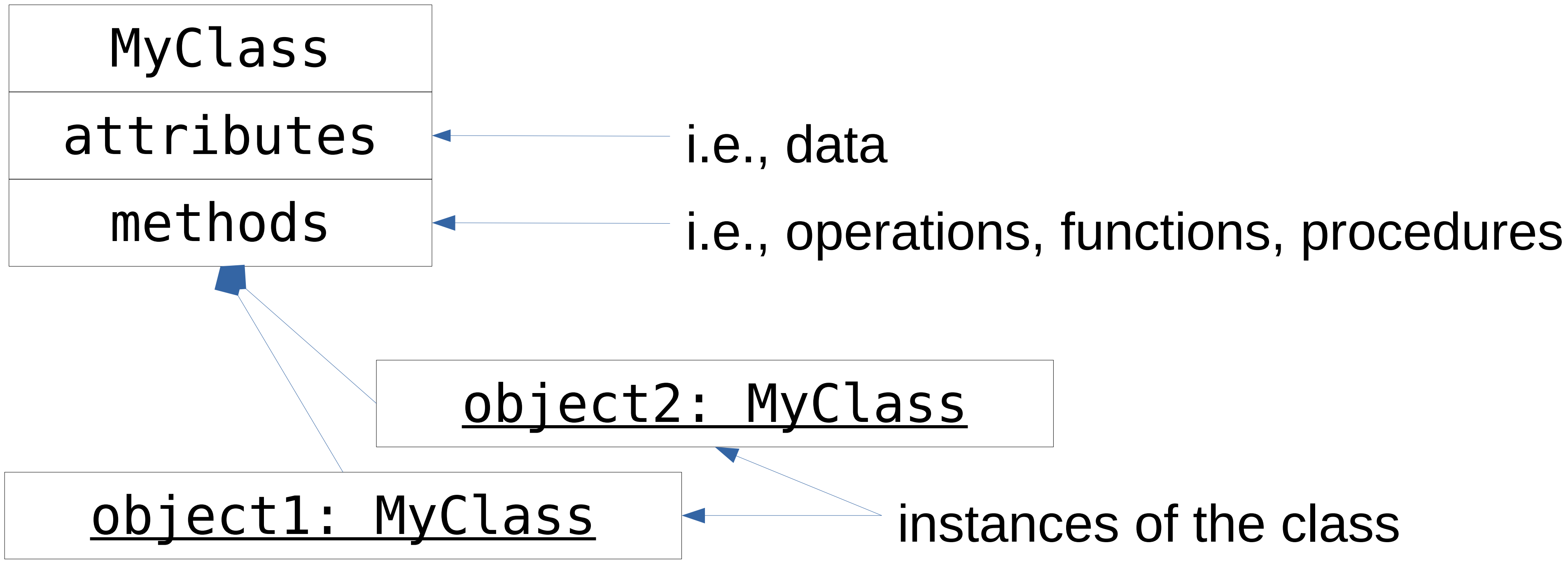
univ-cotedazur.fr

Unit 2
A Crash Course on Object-Oriented Programming in Python

Object-Oriented Programming

- A programming discipline whereby a programmer defines
  - Not only data structure
  - But also the operations that may be applied to them
- This way
  - A data structure become an **object**, which includes
    - Data, called **attributes**
    - Operations, called **methods**
  - The programmer may define **relationships** between the objects

Objects and Classes



MyClass

attributes

methods

i.e., data

i.e., operations, functions, procedures

object2: MyClass

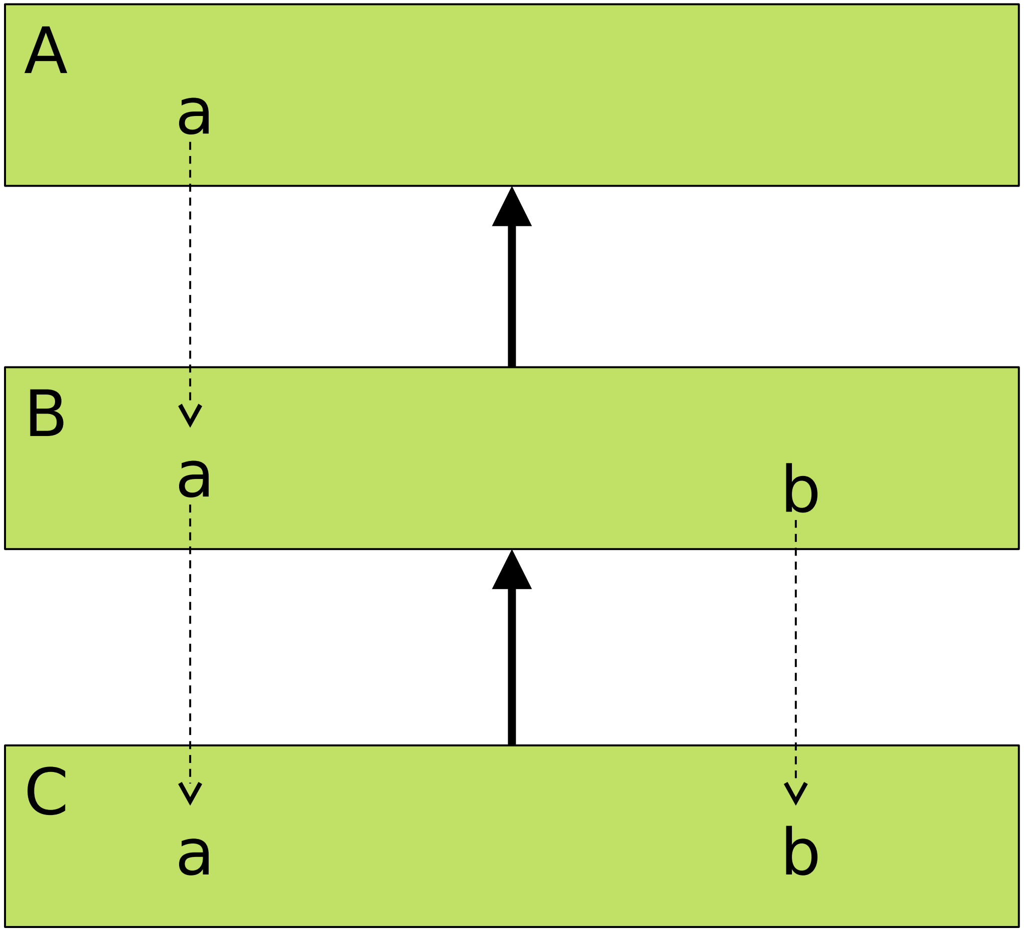
object1: MyClass

instances of the class

Heritage

- Heritage is the characteristic of an object-oriented language whereby the objects of a class “inherit” all the properties defined for the classes of a higher level:
  - Attributes
  - Methods
  - Constants
- Historically, this is one of the most controversial characteristics

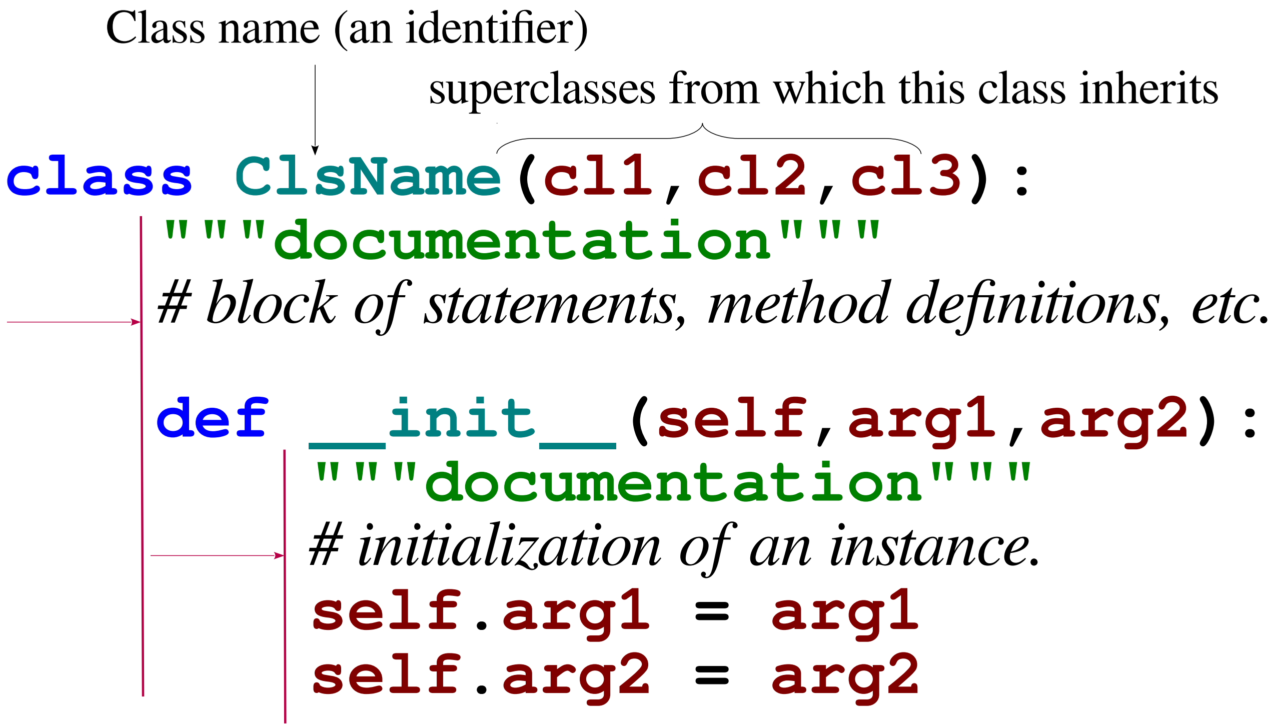
Heritage



OOP in Python

- Objects are the abstraction of data
- All data in a Python program are represented by objects, including code
  - An object possesses:
  - An identity (= its memory address) : id()
  - A type (supported operations + possible values)
  - A value (mutable or immutable, depending on the type)
- Each class and class instance possesses a name space, implemented as a dictionary

Class Definition



Class name (an identifier)

superclasses from which this class inherits

```
class ClsName(cl1,cl2,cl3):
    """documentation"""
    # block of statements, method definitions, etc.

    def __init__(self,arg1,arg2):
        """documentation"""
        # initialization of an instance.
        self.arg1 = arg1
        self.arg2 = arg2
```

Creating and Using Objects

```
instance = ClsName(arg1,arg2)

instance.method(x,y)

instance.attribute

ClsName.class_attribute
```