Object oriented programming

Plan

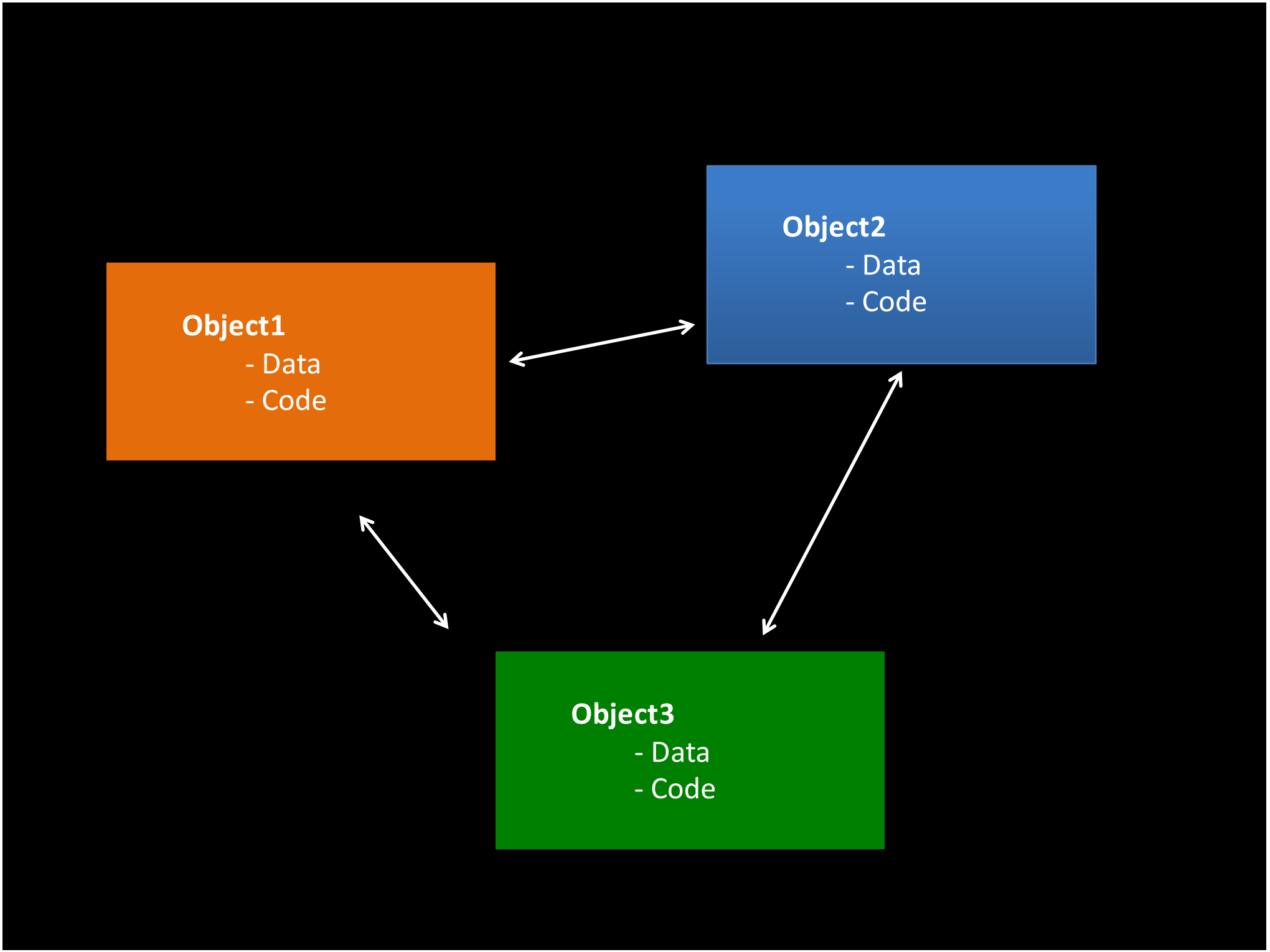
1. What is object oriented programming (OOP)?

2. Exercises to go over the concepts and the syntax associated with them in python.

Procedural programming

- Procedural programming focuses on the series of actions.
- It contains data and actions all mixed together.
- Very linear!

- C, Fortran


Object1
- Data
- Code
Object2
- Data
- Code
Object3
- Data
- Code

Object oriented

- Data before action.
- It is based on objects, objects contain their own data and their own logic.
- Object oriented is really about object sspeaking to each other.
- Python, C++, Java, Perl, Ruby

Classes – type of objects

Time	Date	Project
Gene	Fasta	TelescopeObservation
WeatherData	Temperature	Sample

What defines a class?

WeatherData

Attributes

- Geographic coordinates
- Humidity
- Temperature
- Atmospheric Pressure
- Time
- Date

Methods

- Calculate chance of rain
- Compare to monthly average
- Plot on a map

Objects are instances of Classes

- Classes are description of objects properties.
- There is no point defining a class if you are not going to make an object.

Class: Date

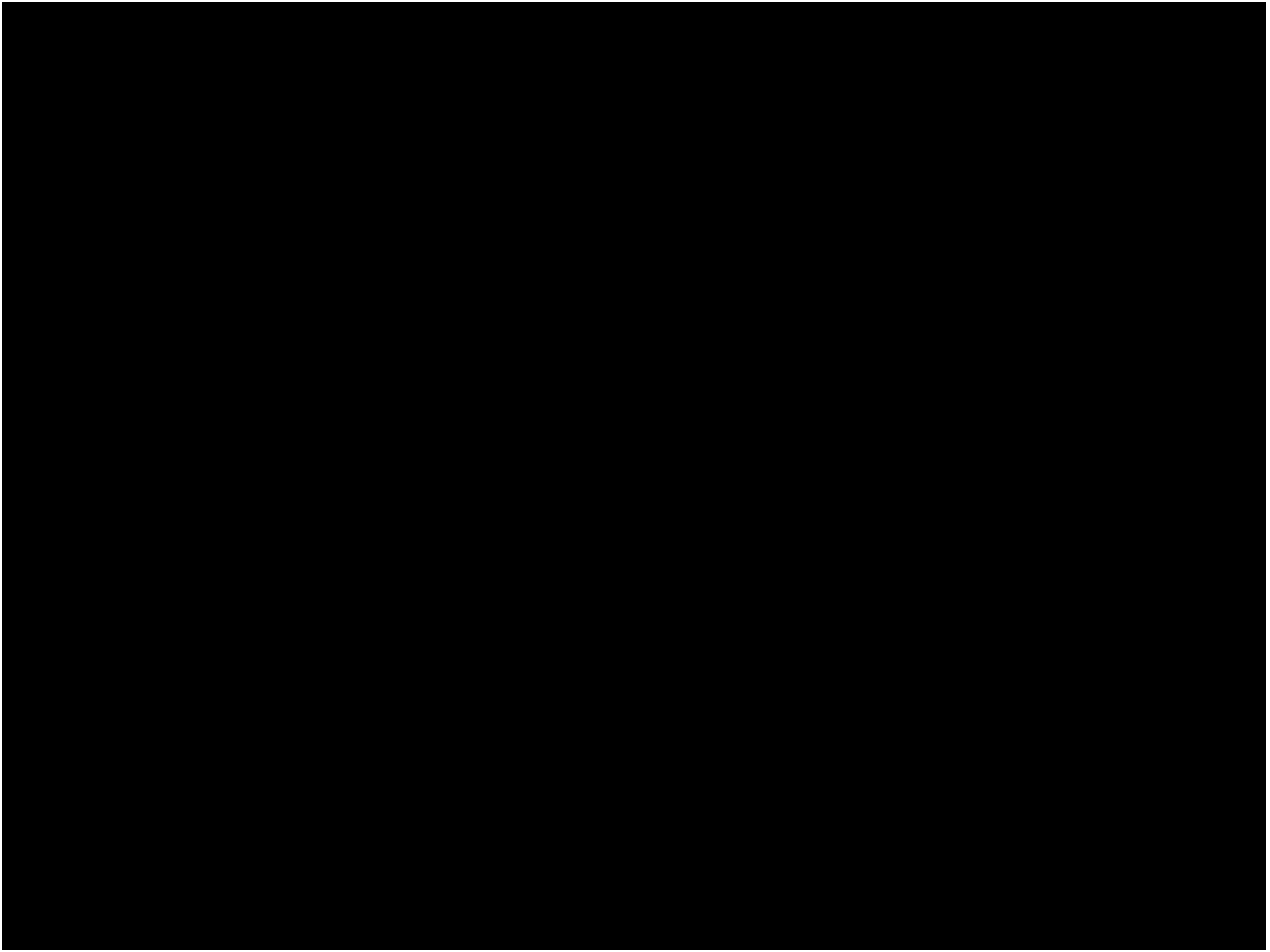
Object: Yesterday Today Tomorrow

Object orienting programming

- It is an idea, a way of seeing code as objects interacting with each other.
- These objects contain both code and data.

Exercise

- Take 5 minutes to think of a class that could related to analyses you are doing in your PhD project (A type of data, observation, file format).
- Give it 2 attributes and 2 methods.
- Think of 1 instance of that class, what would be its attributes, what would happen if you call the method?



Classes relationship: Inheritance (is)

Class	**Attributes**
Vehicule	nbPassengers; color
Aircraft(Vehicle)	fuel type; tank size
Helicopter(Aircraft)	number of helixes

Classes relationship: Composition (has)

WeatherData

- Attributes

- Geographic coordinates
- Humidity
- **Temperature**
- Atmospheric Pressure
- **Time**
- **Date**

Composition VS Inheritance

- A helicopter *is* an aircraft.
- A Weather observation *has* a date but a date is not a Weather observation!

Summary of the theory

- It is an idea
- Object at the center; objects are instances of classes, associated with properties and methods.
- Classes can *inherit* from each other (is-a) or be can *compose* each other (has-a)

```
class Fruit(object):
    """A class that makes various tasty fruits."""
    def __init__(self, name, color, flavor, poisonous): #booting method
        self.name = name
        self.color = color
        self.flavor = flavor
        self.poisonous = poisonous

    def description(self): #method
        print "I'm a %s %s and I taste %s." % (self.color, self.name, self.flavor)

    def is_edible(self):
        if not self.poisonous:
            print "Yep! I'm edible."
        else:
            print "Don't eat me! I am super poisonous."

lemon = Fruit("lemon", "yellow", "sour", False)# instance of lemon

lemon.description()# call the method description on lemon
lemon.is_edible()# call the methon is_edible on lemon
lemon.pH = 1 # it is an instance variable, specific to lemon, not all fruits have pH
```