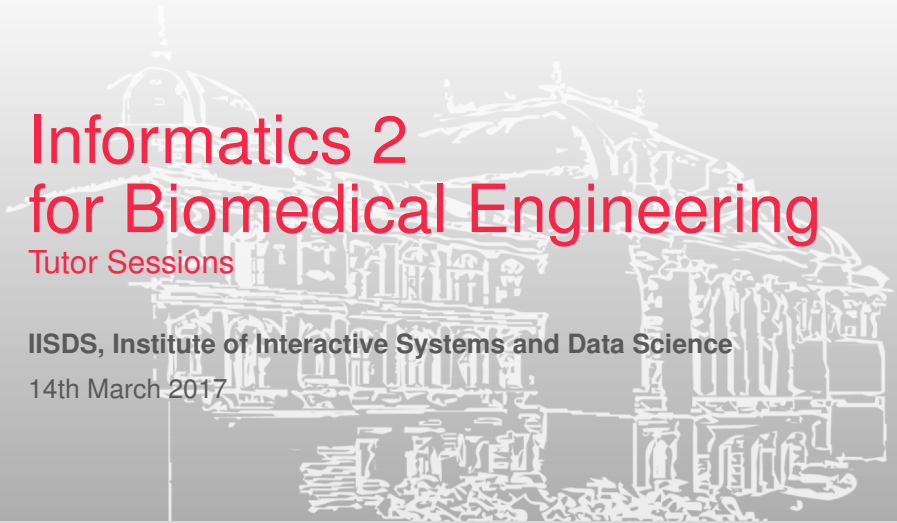

Informatics 2 for Biomedical Engineering

Tutor Sessions

IISDS, Institute of Interactive Systems and Data Science

14th March 2017

Object Oriented Programming

Unit 1

Today's Topics

- Basics of OOP
  - Classes
  - Objects
  - Methods
  - Inheritance
- Creating a Class
- Getter and Setter
- Creating an Instance
- Exercises

Basics of OOP - Classes & Objects

A class...

- ...defines how its objects will be built
- ...has methods and variables

An object...

- ...contains real values (unlike the class)
- ...is an instance of a class

Like a blueprint or recipe (class) and an actual building or a cake (object).

Methods & Inheritance

A method...

- ...is part of a class
- ...of a class will be included in all objects of that class

E.g. a car class/object could have a drive method.

Inheritance...

- ...is when one class gives characteristics to a class that is derived from it

There could be a Mammal class. We then could define a Human or Ape class by subclassing Mammal and would inherit all methods and variables from Mammal.

Creating a Class

```
1   # A simple class
2
3   class FirstClass:
4     instances = 0
5
6     def __init__(self, letters):
7       self.letters = letters
8       FirstClass.instances += 1
9
10    def getLetters(self):
11      return self.letters
12
13  a = FirstClass("asdf")
14  print(a.getLetters())
```

Getter and Setter

- Getter: @property
- Setter: @x.setter
- Deleter: @x.deleter

'x' stands for the member variable.
Use an '_' in front of the variable to mark it as private (Python doesn't restrict its access). E.g. _speed
[1]

[1] https://docs.python.org/3/library/functions.html#property

Getter and Setter - Example

```
1   class SecondClass:
2       def __init__(self, letters):
3           self._letters = letters
4
5       @property
6       def letters(self):
7           return self._letters
8
9       @letters.setter
10      def letters(self, newletters):
11          self._letters = newletters
12
13      @letters.deleter
14      def letters(self):
15          del self._letters
```

Creating an Instance

```
1   classinst1 = SecondClass("Test1")
2   astring = "Test2"
3   classinst2 = SecondClass(astring)
4
5   print(classinst1.letters)
6
7   classinst1.letters = "Test3"
8   print(classinst1.letters)
9
10  del(classinst1.letters)
11  print(classinst1.letters) # What would this print?
```

Exercises - Task 1

Write a class ”Vehicle” with the __init__ function so that it sets the maximum speed and the wheels to numbers that are entered upon instantiation.
Add a ”drive” function that calculates the km from speed and minutes and returns the result. Then add a getter, setter and deleter for kilometres, where the setter adds a km number to the current kilometres. Write getters for maximum speed and wheels.
Then write a second class ”Car” that inherits from the class Vehicle and also sets the number of doors. Write a getter for the number of doors.
Create an object for each class.

Exercises - Task 2

Write a class ”Pet” that takes the name of a pet. Add a getter for the name.
Then write a second class that inherits from Pet, with a name and an attribute of your choice (e.g. ”Cat” with attribute ”scratches” or ”Dog” with attribute ”drools”, etc.). Add a getter and a setter for this attribute.
Create an object for your second class, printing out the pet’s name and your attribute of choice’s value. (Output example: ”Fido”, 1)