PYTHON SEMINAR 2020

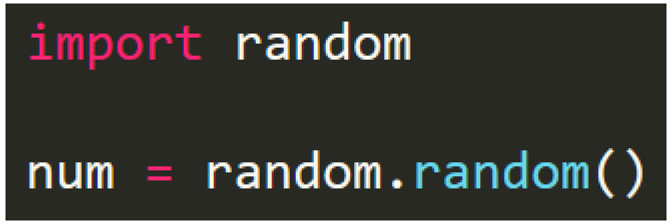
JENS HAHN

THEORETICAL BIOPHYSICS

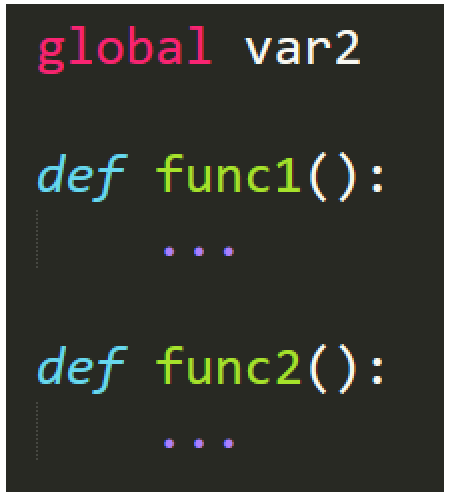
FIRST PROGRAMME

Challenges

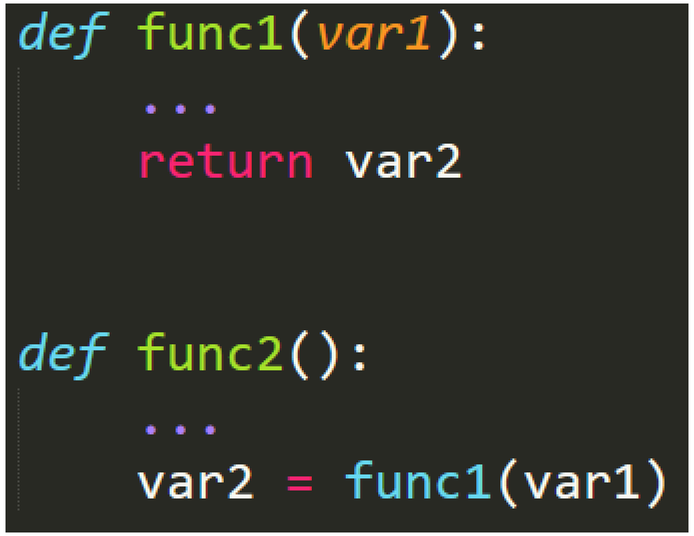
1. Load packages
2. Structure code
    1. Connect several functions
    2. Share variables (`global` vs. arguments)



```python
import random

num = random.random()
```



```python
global var2

def func1():
    ...

def func2():
    ...
```



```python
def func1(var1):
    ...
    return var2


def func2():
    ...
    var2 = func1(var1)
```

TODAY

I How to start programming?

II Recap functions

III Classes & instances

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD ‘YEAST’ IN TEXT FILES

Theory

1. What steps need to be taken?

Example

1. Load text files
2. Parse through them
3. Identify word

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD 'YEAST' IN TEXT FILES

Theory

1. What steps need to be taken?
2. Think about your input/output

Example

- Input: Text file (file name)
- Output: '*Yes*' or '*No*'

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD 'YEAST' IN TEXT FILES

Theory

1. What steps need to be taken?
2. Think about your input/output
3. Do I have the necessary tool?

Example

- Syntax to write a function in Python
- How to open a file
- How to parse through an opened file
- String comparison

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD 'YEAST' IN TEXT FILES

Theory

1. What steps need to be taken?
2. Think about your input/output
3. Do I have the necessary tool?

Example

- Syntax to write a function in Python
- How to open a file
- How to parse through an opened file
- String comparison

➡ Google, StackOverflow, Python docu

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD 'YEAST' IN TEXT FILES

Theory

1. What steps need to be taken?
2. Think about your input/output
3. Do I have the necessary tool?
4. Test the tools separately
   1. Read errors carefully!!

Example

- Open a file in a Python console
- Parse through a long *string* in console
- Compare strings
- Error message gives line number!

I. HOW TO START PROGRAMMING?

LOOK FOR THE WORD 'YEAST' IN TEXT FILES

Theory

1. What steps need to be taken?
2. Think about your input/output
3. Do I have the necessary tool?
4. Test the tools separately
   1. Read errors carefully!!
5. Combine steps
   1. Use print to ensure correctness

Example

- Is the file read correctly?
- Do you miss any lines when parsing?
- Is the comparison working?
  - Spaces?
  - Punctuation?
  - Encoding?

II. RECAP FUNCTIONS

- Functions are small (!) code blocks with one (!) task
- Functions can be re-used easily
- Functions can have arguments
- Functions have their own namespace
- Functions always return something

I. ERROR MESSAGES

```
NameError: name 'meep' is not defined
```

```
my_var = meep
```

```
IndentationError: unexpected indent
```

```
def meep():
    a = 3
      b=4
```

```
SyntaxError: invalid syntax
```

```
def meep:
```

```
IndexError: list index out of range
```

```
my_list[1]
```

```
TypeError: 'list' object is not callable
```

```
my_list()
```

```
TypeError: list indices must be integers or slices, not str
```

```
my_list['meep']
```

```
TypeError: 'int' object is not iterable
```

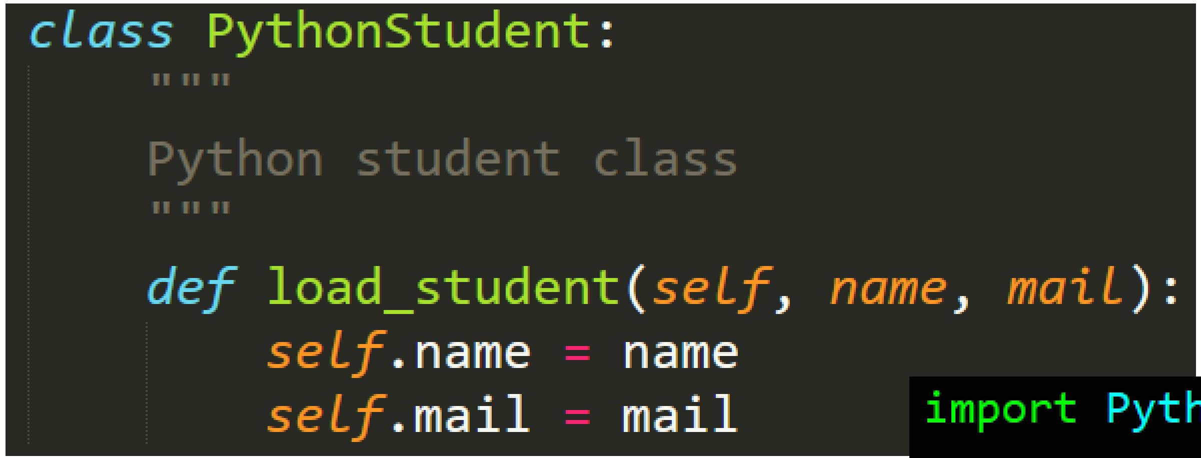
```
for i in len(my_list):
```

III. CLASSES

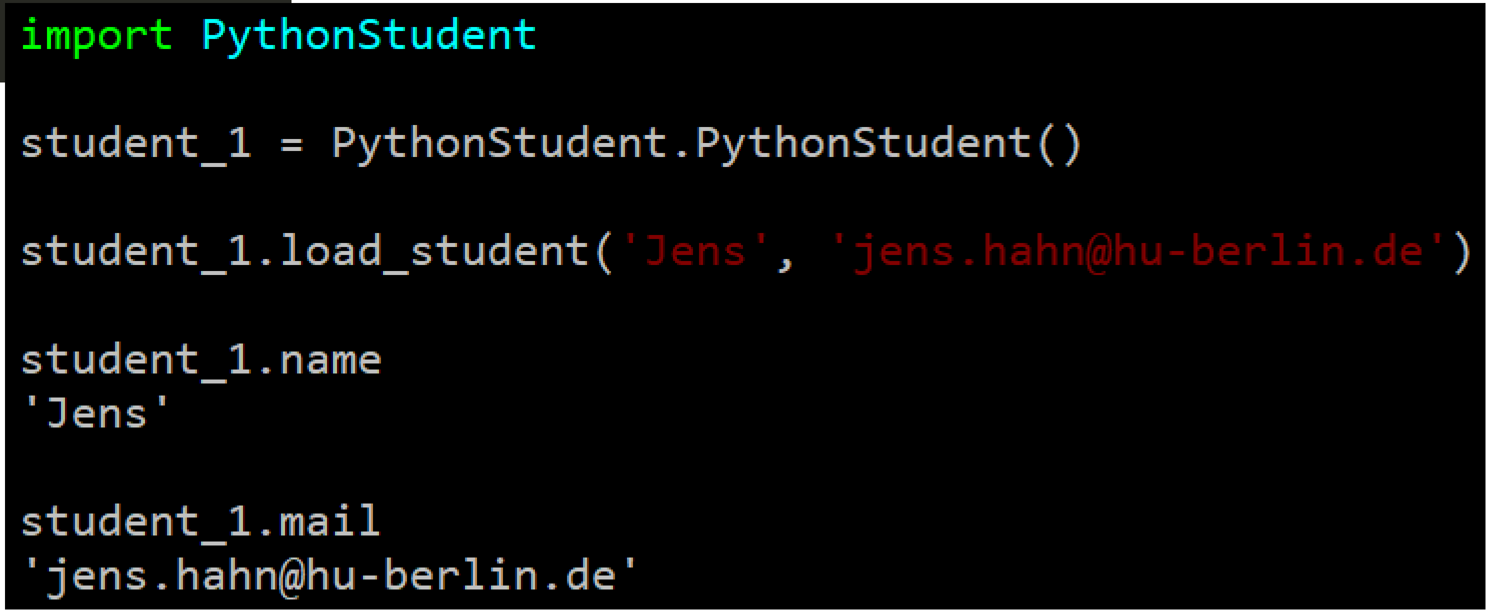
- Everything in Python is an *object* but we didn't made one yet!
- Remember: classes have methods (functions) and class variables
- We need to declare class attribute (class scope)
    - Everything combined with ***self*** will be available from the class:

```
self.my_variable = []                def my_mehthod(self):
```

III. CLASSES



```
class PythonStudent:
    """
    Python student class
    """
    def load_student(self, name, mail):
        self.name = name
        self.mail = mail
```



```
import PythonStudent

student_1 = PythonStudent.PythonStudent()

student_1.load_student('Jens', 'jens.hahn@hu-berlin.de')

student_1.name
'Jens'

student_1.mail
'jens.hahn@hu-berlin.de'
```

III. CLASSES - THE MAGIC METHODS

- Some methods are always set automatically (inherited from *object*)
- **`__init__()`** : Initialization method (called always)
- **`__dir__()`** : Show all available methods and attributes
- **`__repr__()`** : Official *string* representation of *object*
- **`__doc__()`** : Show the doc string of the class (documentation)

V. EXAMPLE - FRACTION OBJECTS

Create a fraction data type

- Data type stores fractions (1/2, 3/4,…)
- Addition, subtraction, multiplication, division (overloading)
- Reduction of fractions

V. FURTHER READING

Python classes

- Programiz – Python Classes

  https://www.programiz.com/python-programming/class

- Python class - documentation

  https://docs.python.org/3/tutorial/classes.html

Python inheritance and magic methods

- Programiz - Python inheritance

  https://www.programiz.com/python-programming/inheritance

- *Tutorials Teacher – magic methods*

  https://www.tutorialsteacher.com/python/magic-methods-in-python