Python Seminar 2018

Ana Bulović
Jorin Diemer
Jens Hahn

Plan for today

1. Homework
2. Class usage example
3. Overloading
4. Coding

Homework

Python magic functions

- Used for implementing elegant object access, comparison, conversion etc.
- they are surrounded by two underscores (known example: __init__)

If we write:

```
if a == b:
    print ("They are equal, 'tis true!")
```

What is executed is:

```
if a.__eq__(b):
    print ("They are equal, 'tis true!")
```

Meaning that object *a* has to implement the __eq__ function!

The same goes for printing:

```
print(a)  ⟶  print(a.__str__())
```

```python
from functools import total_ordering


@total_ordering
class Metabolite(object):
    def __init__(self, name, compartment, concentration=0):
        self.name = name
        self.compartment = compartment
        self.concentration = concentration

    def __eq__(self, other):
        if self.name == other.name and self.compartment == other.compartment \
           and self.concentration == other.concentration:
            return True
        else:
            return False

    def __gt__(self, other):
        if self.concentration > other.concentration:
            return True
        else:
            return False

    def __repr__(self):
        return "Metabolite('{}', '{}', {})".format(self.name,
                                                   self.compartment,
                                                   self.concentration)

    def __str__(self):
        return "{}({})  (C = {})".format(self.name, self.compartment, self.concentration)
```

```
In [3]: m1 = Metabolite('ATP', 'c', 2.8)
   ...: m2 = Metabolite('ADP', 'c', 1.3)
   ...: m3 = Metabolite('glc', 'e', 5)
   ...:

In [4]: m1
Out[4]: Metabolite('ATP', 'c', 2.8)

In [5]: print(m1)
ATP(c)  (C = 2.8)

In [6]: m1 == m2
Out[6]: False

In [7]: m1 > m2
Out[7]: True

In [8]: sorted([m1, m2, m3])
Out[8]:
[Metabolite('ADP', 'c', 1.3),
 Metabolite('ATP', 'c', 2.8),
 Metabolite('glc', 'e', 5)]
```

Magic function categories

- Construction and initialization (__new__, __init__, __del__)
- Comparison (__eq__, __ne__, __lt__, __gt__, __le__, __ge__)
- Numeric operations: unitary (__pos__, __neg__, __abs__, …) and binary (__add__, __sub__, __mul__, …)
- Type conversion(__int__, __long__, __float__, …)
- Class representation (__str__, __repr__, __format__)
- Attribute access control (__getattr__, __setattr__)

Agent-based modelling

- Independent agents

- Many possibilities
  - supermarkets
  - emergency exits
  - lipid metabolism
  - sperm movement

HOMEWORK

Build your own small agent-based model!!