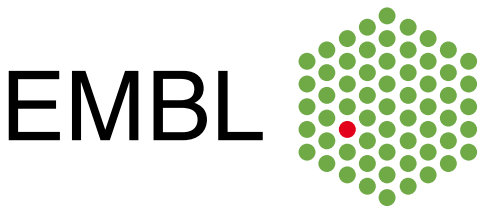
More Python Types & Functions

Luis Pedro Coelho
§
On the web: http://luispedro.org
On twitter: @luispedrocoelho

European Molecular Biology Laboratory

June 10, 2014



Set Type

```
numbers = set([1,2,5])
print 3 in numbers
numbers.add(4)
print numbers
numbers.add(1)
print numbers
print numbers | set(['Rita'])
print numbers - set([2,3])
```

Output:

```
False
set([1, 2, 4, 5])
set([1, 2, 4, 5])
set([1, 2, 4, 5, 'Rita'])
set([1, 4, 5])
```

None object

None

Object Identity

Object Identity

- A is B
- A is not B

Exercise

```
A = []
B = []
A.append(1)
B.append(1)

print (A == B)
print (A is B)
```

This prints:

(a)	(b)	(c)	(d)
True True	False True	False False	True False

Exercise Break

Consider the following code:

```
g2g = {
        'PBANKA_000230': ['GO:0003899'],
        'PBANKA_000370': ['GO:0016740'],
        'PBANKA_010060': ['GO:0030430'],
        'PBANKA_010080': ['GO:0008270'],
}
```

(In real life, this would have 2420 entries)

Exercise Break

Consider the following code:

```
g2g = {
      'PBANKA_000230': ['GO:0003899'],
      'PBANKA_000370': ['GO:0016740'],
      'PBANKA_010060': ['GO:0030430'],
      'PBANKA_010080': ['GO:0008270'],
}
```

(In real life, this would have 2420 entries)
How do you look up GO term for gene PBANKA_000230?

Exercise Break

Consider the following code:

```
g2g = {
      'PBANKA_000230': ['GO:0003899'],
      'PBANKA_000370': ['GO:0016740'],
      'PBANKA_010060': ['GO:0030430'],
      'PBANKA_010080': ['GO:0008270'],
}
```

(In real life, this would have 2420 entries)
How do you look up GO term for gene PBANKA_000230?

(a)	(b)	(c)
g2g[0]	g2g['PBANKA_000230']	g2g[000230]

List Comprehensions

```
name = [ <expr> for <name> in <sequence> if <condition> ]
```

maps to

```
name = []
for <name> in <sequence>:
    if <condition>:
        name.append(<expr>)
```

List Comprehensions Example

```
squares = [x*x for x in xrange(1,20)]

squares = []
for x in xrange(1,20):
    squares.append(x*x)
```

Functions I

```
def greet():
    print 'Hello World'
    print 'Still Here'

greet()
greet()
print 'Now here'
greet()
```

Functions II

```
def greet(name):
    print 'Hello {0}'.format(name)

greet('World')
greet('Luis')
greet('Kim')
```

Functions III

```
def max(xs):
    '''
    M = max(xs)

    Returns the maximum of ``xs``
    '''
    M = xs[0]
    for x in xs[1:]
        if x > M:
            M = x
    return M
```

Multiple Assignment

```
A, B = 1, 2
```

Assign multiple elements at once.

```
def greet(name, greeting='Hello'):
    '''
    greet(name, greeting='Hello')

    Greets person by name

    Parameters
    ----------
    name: str
        Name
    greeting: str, optional
        Greeting to use
    '''
    print greeting, name

ret = greet('World')
```

Sequences

```
for value in sequence:
    ...
```

Sequences

- Lists
- Tuples
- Sets
- Dictionaries
- ...

Goals for next 15 minutes

- A quiz
- Do a few exercises.
- Play around.
- You can work alone, in pairs, in triples,...
- Looking up answers on the internet is technique, not cheating!

Lists I

How do you access the first element of a list?
Assume list is a list:

1. list[1]
2. list[0]
3. list[-1]
4. list(0)
5. list(-1)
6. list(1)

Lists II

How do you access the last element of a list?
Assume list is a list:

1. list[1]
2. list(-0)
3. list[-1]
4. list(-1)
5. list(1)
6. list[-0]

Exercises

Object Identity

What is the difference between the following two code examples:
A)

```
A = [1, 2, 3]
B = [1, 2, 3]
```

B)

```
A = [1, 2, 3]
B = A
```

Write a small piece of code (should be 2 or 3 lines) that behaves differently if you insert it after each of the two segments above.

Object Identity

What is the difference between the following two code examples:
A)

```
A = [1, 2, 3]
B = [1, 2, 3]
```

B)

```
A = [1, 2, 3]
B = A
```

Write a small piece of code (should be 2 or 3 lines) that behaves differently if you insert it after each of the two segments above.

```
B[0] = 0
print A
```

sum

1. Learn about the built-in function sum
2. Write an implementation of this function

sum

1. Learn about the built-in function sum
2. Write an implementation of this function

```
def sum(xs, start=0):
    '''
    s = sum(xs, start=0)

    Returns the sum of all values in ``xs`` + ``start``
            (which defaults to 0)
    '''
    for x in xs:
        start += x
    return start
```

```
numbers = set([1,2])
for i in xrange(5):
    numbers.add(i)
print len(numbers)
```

This prints:

- 7
- 6
- 5
- 4

Learning more

- Learn Python the Hard Way by Zed Shaw
  (online for free or pay money for hard copy)
- http://python.org

Numpy Historical

- Numeric (1995)
- Numarray (for large arrays)
- scipy.core (briefly, around 2005)
- numpy (2005)

Currently

- numpy 1.6
- de facto standard
- very stable

Basic Type

numpy.array or numpy.ndarray.

Multi-dimensional array of numbers.

numpy example

```
import numpy as np
A = np.array([
      [0,1,2],
      [2,3,4],
      [4,5,6],
      [6,7,8]])
print A[0,0]
print A[0,1]
print A[1,0]
```

Some Array Properties

```
import numpy as np
A = np.array([
      [0,1,2],
      [2,3,4],
      [4,5,6],
      [6,7,8]])
print A.shape
print A.size
```

Some Array Functions

```
...
print A.max()
print A.min()
```

- max(): maximum
- min(): minimum
- ptp(): spread (max - min)
- sum(): sum
- std(): standard deviation
- ...

Other Functions

- np.exp
- np.sin
- ...

All of these work element-wise!

Arithmetic Operations

```
import numpy as np
A = np.array([0,1,2,3])
B = np.array([1,1,2,2])

print A + B
print A * B
print A / B
```

Broadcasting

Mixing arrays of different dimensions

```
import numpy as np
A = np.array([
      [0,0,1],
      [1,1,2],
      [1,2,2],
      [3,2,2]
      ])
B = np.array([2,1,2])

print A + B
print A * B
```

Broadcasting

Special case: scalar.

```
import numpy as np
A = np.arange(100)
print A + 2
A += 2
```

Data Types

numpy.ndarray is a homogeneous array of numbers.

Types

- Boolean
- integers
- floating point numbers
- ...

Object Construction

```
import numpy as np
A = np.array([0,1,1],float)
A = np.array([0,1,1],bool)
```

Reduction

```
A = np.array([
      [0,0,1],
      [1,2,3],
      [2,4,2],
      [1,0,1]])
print A.max(0)
print A.max(1)
print A.max()
```

prints

```
[2,4,3]
[1,3,4,1]
4
```

The same is true for many other functions.

Slicing

```
import numpy as np
A = np.array([
      [0,1,2],
      [2,3,4],
      [4,5,6],
      [6,7,8]])
print A[0]
print A[0].shape
print A[1]
print A[:,2]
```

Two minute break

- Talk to your neighbours
- Play around in Python
- Ask questions

Slices Share Memory!

```
import numpy as np
A = np.array([
      [0,1,2],
      [2,3,4],
      [4,5,6],
      [6,7,8]])
B = A[0]
B[0] = -1
print A[0,0]
```

Pass is By Reference

```
def double(A):
    A *= 2

A = np.arange(20)
double(A)
```

Pass is By Reference

```
def double(A):
    A *= 2

A = np.arange(20)
double(A)

A = np.arange(20)
B = A.copy()
```

Logical Arrays

```
A = np.array([-1,0,1,2,-2,3,4,-2])
print (A > 0)
```

Logical Arrays II

```
A = np.array([-1,0,1,2,-2,3,4,-2])
print ( (A > 0) & (A < 3) ).mean()
```

What does this do?

Logical Indexing

```
A[A < 0] = 0
```

or

```
A *= (A > 0)
```

Logical Indexing

```
print 'Mean of positives', A[A > 0].mean()
```

Some Helper Functions

Constructing Arrays

```
A = np.zeros((10,10), int)
B = np.ones(10)
C = np.arange(100).reshape((10,10))
...
```

Multiple Dimensions

```
img = np.zeros((1024,1024,3))
```

Documentation

http://docs.scipy.org/doc/

Matplotlib

Matplotlib is a plotting library for Python.

```
import pylab
import numpy as np
X = np.linspace(-4,+4,1000)
pylab.plot(X,np.exp(-X**2))
pylab.xlabel(r'$x$')
pylab.ylabel(r'$\exp(-x^{2})$')
pylab.savefig('gaussian.pdf')
```

http://matplotlib.sf.net/

Matplotlib Example