More Python Types & Functions

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

European Molecular Biology Laboratory

April 3, 2014
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

- 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:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Consider the following code:

```python
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)
Consider the following code:

```
max = {
    '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_00230?

(a) `g2g[0]`

(b) `g2g['PBANKA_00230']`

(c) `g2g[00230]`
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_00230?

(a) \(g2g[0]\)  
(b) \(g2g['PBANKA_00230']\)  
(c) \(g2g[00230]\)
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)
```
def greet():
    print 'Hello World'
    print 'Still Here'

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

greet('World')
greet('Luis')
greet('Kim')
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')
for value in sequence:
...

Sequences

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

...
Goals for next 30 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!
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)`
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
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.
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
1. Learn about the built-in function `sum`
2. Write an implementation of this function
Learn about the built-in function `sum`

Write an implementation of this function

```python
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
Learn Python the Hard Way by Zed Shaw
(online for free or pay money for hard copy)

http://python.org