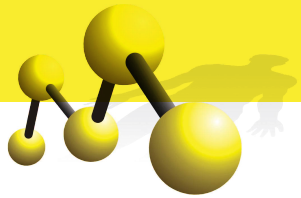
More Python Types & Functions

Luis Pedro Coelho

Programming for Scientists

October 22, 2012

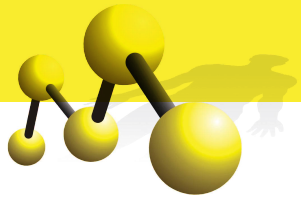
Python So Far



Python

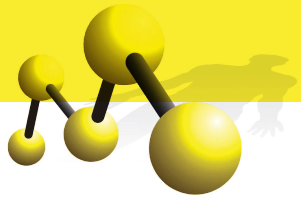
1. Basic types: int, float, list
2. Control flow: for, while, if, else, elif

List Indexing



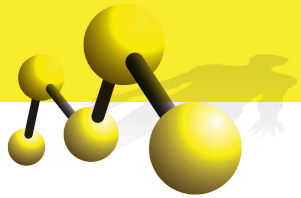
```
students = ['Luis','Rita','Sabah','Grace']
print students[0]
print students[1:2]
print students[1:]
print students[-1]
print students[-2]
```

Tuples (I)



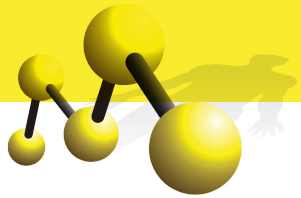
```
A = (0,1,2)
B = (1,)

print A[0]
print len(B)
```

Tuples (II)



Tuples are like immutable lists.

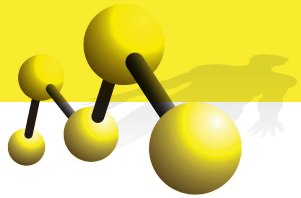
Dictionaries



- Dictionaries are associative arrays.

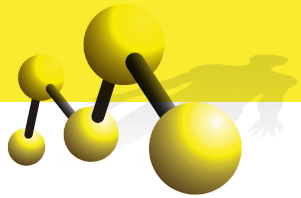
```
gene2ensembl = {}
gene2ensembl['SMAD9'] = 'ENSG00000120693'
gene2ensembl['ZNF670'] = 'ENSG00000135747'

print gene2ensembl['SMAD9']
```

Dictionary Methods



```
gene2expression = {
      'SMAD9' : 12.3,
      'ZNF670' : 4.3,
}

print len(gene2ensembl)
print gene2ensembl.keys()
```

Set Type

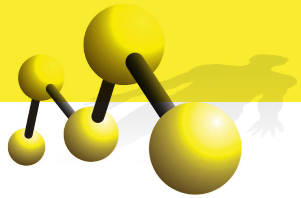


```
numbers = set([1,2,5])
print 3 in numbers
numbers.add(4)
print numbers
numbers.add(1)
print numbers
print numbers | set(['Rita'])
print numbers - set([2,3])
```

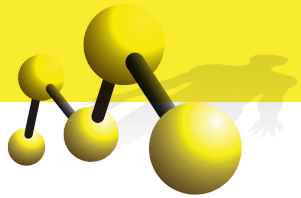
Output:

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

None object



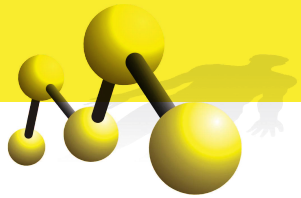
None



Object Identity

- A is B
- A is not B

Exercise

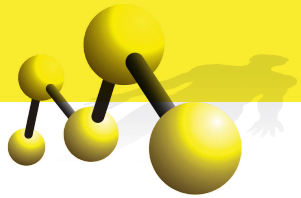


```
A = []
B = []
A.append(1)
B.append(1)

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

This prints:

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

Exercise Break

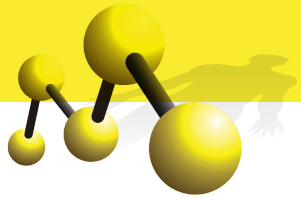


Consider the following code:

```
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

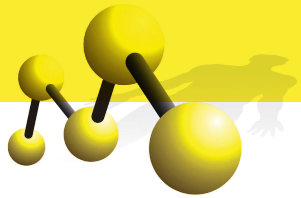


Consider the following code:

```
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_00230?

Exercise Break



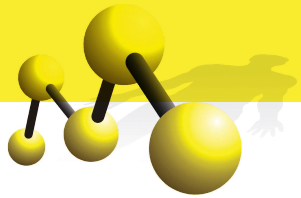
Consider the following code:

```
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_00230?

(a)	(b)	(c)
g2g[0]	g2g['PBANKA_00230']	g2g[00230]

List Comprehensions

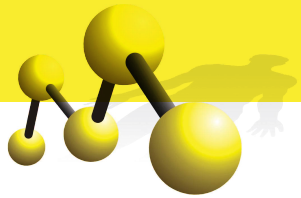


```
name = [ <expr> for <name> in <sequence> if <condition> ]
```

maps to

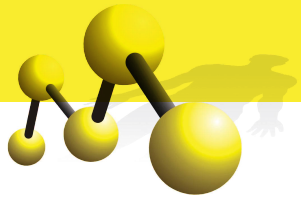
```
name = []
for <name> in <sequence>:
    if <condition>:
        name.append(<expr>)
```

List Comprehensions Example



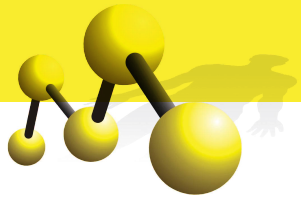
```
squares = [x*x for x in xrange(1,20)]
evensquares = [x*x for x in xrange(1,20) if (x%2) == 0]

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

evensquares = []
for x in xrange(1,20):
    if (x%2) == 0:
        evensquares.append(x*x)
```

Functions I



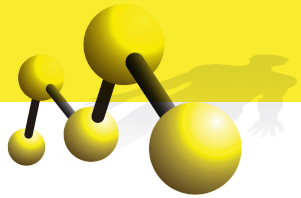
```
def greet():
    print 'Hello World'
    print 'Still Here'

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

Functions II



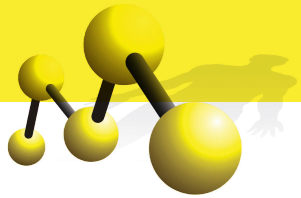
```
def greet(name):
    print 'Hello {0}'.format(name)

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

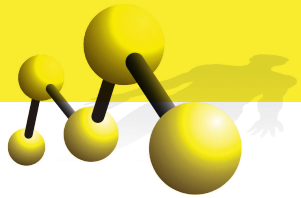


```
A, B = 1, 2
```

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
- ...