Python and NumPy

Hanock Kwak

2016-09-27

Biointelligence Lab

Computer Science and Engineering, Seoul National University

No Declaration

```
a = 1
print a
```

```
1
```

```
print b
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-ab3a5d8f1075> in <module>()
----> 1 print b

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

Everything is an object

```
a = 1
print a
a = 'abc'
print a
a = [1, 2, 3]
print a
```

```
1
abc
[1, 2, 3]
```

Numbers are immutable

```
a = 1
b = a
b += a
print a, b
```

```
1 2
```

Strings

```
a = 'hello'
print a
b = a + ' ' + 'world'
print b
print b.index('o')
print b.count('o')
print b[6:8]
print b[6:]
print b[0:10:2]
print b*2
```

```
hello
hello world
4
2
wo
world
hlowr
hello worldhello world
```

```
print 'There are %d bananas' % 3
print 'There are %d bananas and %d apples' % (3, 10)
print 'There are {} bananas and {} apples'.format(7, 5)
```

```
There are 3 bananas
There are 3 bananas and 10 apples
There are 7 bananas and 5 apples
```

```
x = 'abc'
y = x
y += 'd'
print x, y
print len(x)
print str(123)
```

```
abc abcd
3
123
```

Lists

```
a = [1, 'any', -3.0]
b = ['object', 'can', ['be', 'added'], 'to', 'list']
c = a + b
print c
print c[2:4]
print c[0] + c[2], c[-3][1]
x = a
x += ['lists are mutable']
print a
print x
```

```
[1, 'any', -3.0, 'object', 'can', ['be', 'added'], 'to', 'list']
[-3.0, 'object']
-2.0 added
[1, 'any', -3.0, 'lists are mutable']
[1, 'any', -3.0, 'lists are mutable']
```

Tuples

- Tuples are immutable versions of lists
- One strange point is the format to make a tuple with one element

```
x = 10
a = (1, 2, x)
print a
b = a
b += ('abc',)
print a, b
```

```
(1, 2, 10)
(1, 2, 10) (1, 2, 10, 'abc')
```

Dictionaries

- Key-Value structure

```
a = {1 : 'hello', 'two' : 42, 'blah' : [1,2,3]}
print a[1], a['two']
print 'two' in a
a['two'] = 2
print a['two']
a['three'] = 3
del a[1]
del a['blah']
print a
```

```
hello 42
True
2
{'three': 3, 'two': 2}
```

If

```
import random

a = random.random() # random value 0 ~ 1
b = random.random()
print a, b

if a < 0.3 and b >= 0.8:
    s = 'pororo'
elif a > b or a > 0.9:
    s = 'pikachu'
elif not a*b < 0.5:
    s = 'elsa'
else:
    s = 'stitch'

print s
print len(s) <= 4
```

```
0.265674050893 0.936769174654
pororo
False
```

For

```
my_list = []
for i in xrange(10):
    my_list.append(i)
print my_list
print [v*2 for v in my_list]
print {'key is ' + str(v): 'value is ' + str(v**2) for v in my_list[4:7]}
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
{'key is 6': 'value is 36', 'key is 5': 'value is 25', 'key is 4': 'value is 16'}
```

Functions

```
def sum_over(some_list):
    a = 0
    for v in some_list:
        a += v
    return a

print [k for k in xrange(10, 20)]
print sum_over(xrange(10, 20))

def what_is_happening():
    print a

what_is_happening()
```

```
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
145
19
```

Functions (cont.)

```
def f(a, b = 2, c = 3):
    return a + b + c

print f(1)
print f(1, -1)
print f('a', 'b', 'c')
```

```
6
3
abc
```

```
def fibo(n, fin):
    if fin(n):
        return 1
    else:
        return fibo(n - 1, fin) + fibo(n - 2, fin)

def fin(n):
    return n <= 1

print fibo(10, fin)
```

```
89
```

Lambda

```
f = lambda x,y : x + y
print f(2, 3)

foo = [2, 18, 9, 22, 17, 24, 8, 12, 27]
print filter(lambda x: x % 3 == 0, foo)

print map(lambda w: len(w), 'It is raining cats and dogs'.split())
```

```
5
[18, 9, 24, 12, 27]
[2, 2, 7, 4, 3, 4]
```

With

```
#set things up
f = open("x.txt")
try:
    # do something with data
    f.close()
finally:
    f.close()
```

```
with open("x.txt") as f:
    data = f.read()
    # do something with data
```

NumPy

- NumPy is the fundamental package for scientific computing with Python. It contains among other things:
  - a powerful N-dimensional array object
  - sophisticated (broadcasting) functions
  - useful linear algebra, Fourier transform, and random number capabilities

Basic

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[1, 0], [0, 1]]) # identity matrix
print A.shape
print A.ndim
print A + B - 1
print A*B
print np.dot(A, B)
```

```
(2, 2)
2
[[1 1]
 [2 4]]
[[1 0]
 [0 4]]
[[1 2]
 [3 4]]
```

Broadcasting

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
v = np.array([1, 2]) # vector
print v.shape
print A + v
print np.log(A)
```

```
(2,)
[[2 4]
 [4 6]]
[[ 0.          0.69314718]
 [ 1.09861229  1.38629436]]
```

Indexing

```
import numpy as np

v = np.arange(10)*3
print v[1]
print v[[1]]
print v[[0, 2, 4]]
idx = np.arange(10)
np.random.shuffle(idx)
print v[idx[0:3]] # randomly selects three values in 'v'
```

```
3
[3]
[ 0  6 12]
[ 3  9 21]
```

Indexing

```
import numpy as np

A = np.arange(100).reshape(4, 5, 5)
print A[0]
print A[0, 0:4, 0:2]
print A[0, 2, :]
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]
 [15 16 17 18 19]
 [20 21 22 23 24]]
[[ 0  1]
 [ 5  6]
 [10 11]
 [15 16]]
[10 11 12 13 14]
```

Indexing with Boolean Arrays

```
import numpy as np

a = np.arange(6).reshape(2,3)
b = a > 3
print b
print a[b]

a[b] = -1
print a
```

```
[[False False False]
 [False  True  True]]
[4 5]
[[ 0  1  2]
 [ 3 -1 -1]]
```

Sum

```
import numpy as np

A = np.arange(24).reshape(2, 3, 4)
print A.sum()
print A.sum(axis=0)
print A.sum(axis=1)
print A.sum(axis=2)
```

```
276
[[12 14 16 18]
 [20 22 24 26]
 [28 30 32 34]]
[[12 15 18 21]
 [48 51 54 57]]
[[ 6 22 38]
 [54 70 86]]
```

```
import numpy as np

A = np.arange(24).reshape(2, 3, 4)
print A[:, 0, 0]
print A[0, :, 0]
print A[0, 0, :]
```

```
[ 0 12]
[0 4 8]
[0 1 2 3]
```

There are so many things to learn!!

- Good Luck! ☺