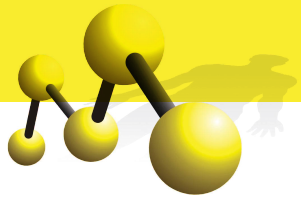
Python IV

Luis Pedro Coelho

Programming for Scientists

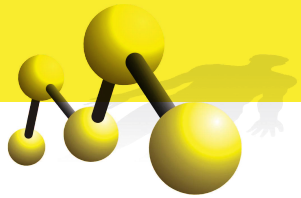
October 22, 2012



Exceptions

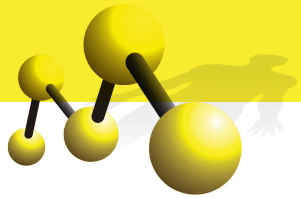
Report errors for higher up.

Call Stack



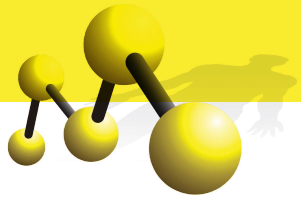
```
def f(x):
    return log(x)**2

def g(x):
    y = f(x)
    return y+1

def h(x):
    return g(x+1) + g(4*x)

print h(0)
```

Exceptions



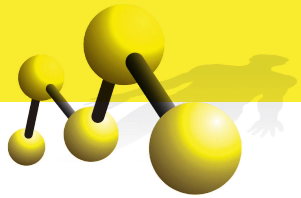
```
def log(x):
    if x <= 0.:
        raise ValueError(
            'log: argument must be greater than zero')
    ...
```

Try-Except



```
try:
    h(0)
except:
    print 'Ooops'
```

Try-Except



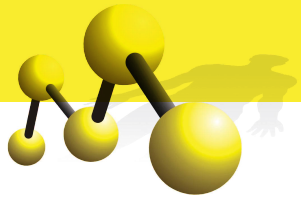
```
try:
    <line 1>
    <line 2>
    <line 3>
except:
    <line 1>
    <line 2>
```

```
def f(x):
    if x <= 0.:
        raise ValueError(
            'f: argument must be greater than zero')
    return sqrt(x)+2

def g(x):
    y = f(x)
    print (y > 2)

try:
    g(1)
    g(-1)
except:
    print 'Exception'
```

This outputs:

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

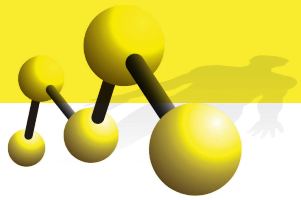
Standard Library Miscellanea



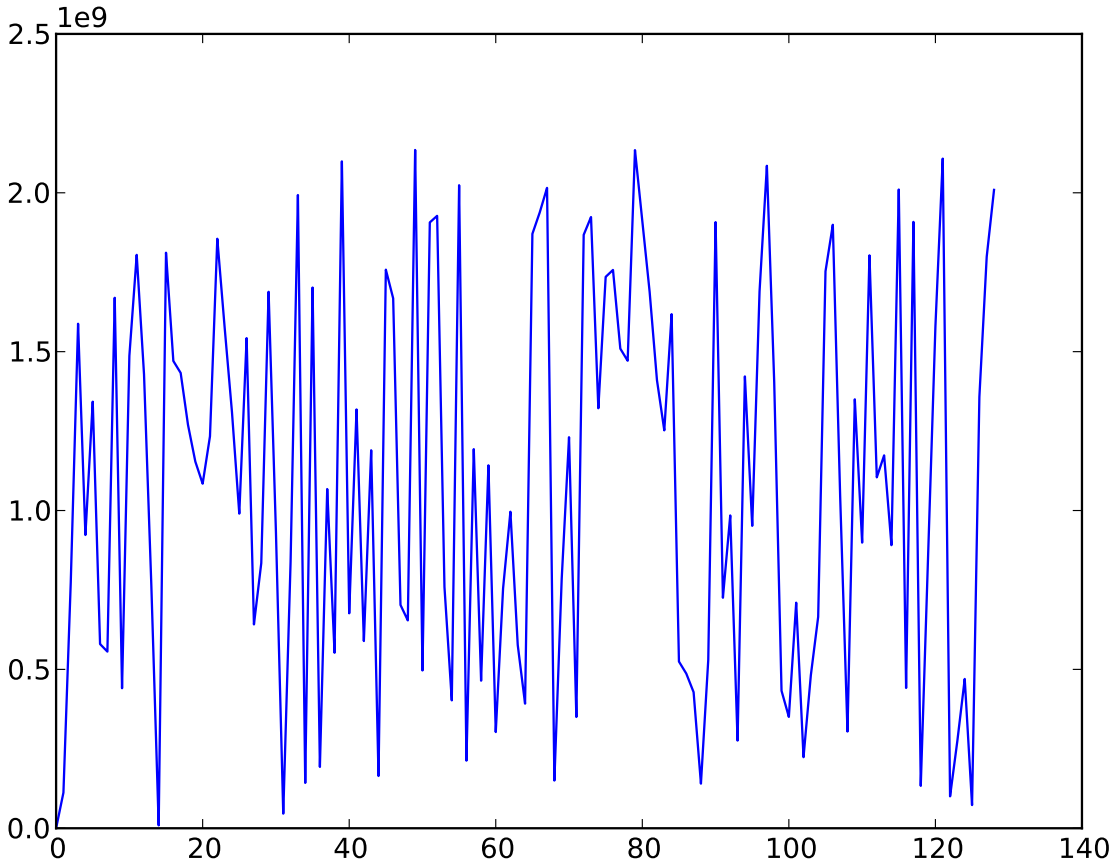
Random numbers

- Truly random numbers
- Pseudo random numbers

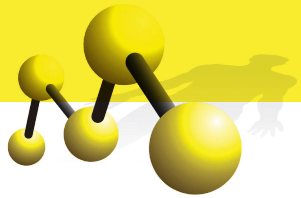
Pseudo Random Numbers



$$x_{i+1} = 48271 x_i \mod (2^{31} - 1)$$

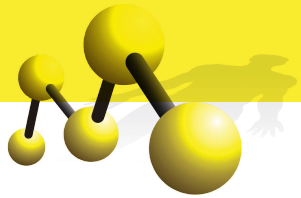


Pseudo Random Numbers



- Are not random
- Some are “more random” than others

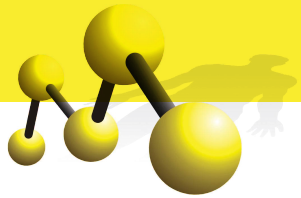
Pseudo Random Numbers



- Are not random
- Some are “more random” than others

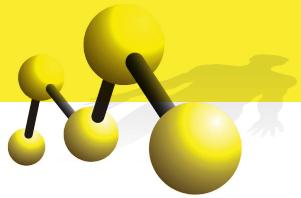
- For testing/reproducibility, you want pseudo-random numbers.
- For cryptography, you want really random numbers.

Testing with random numbers



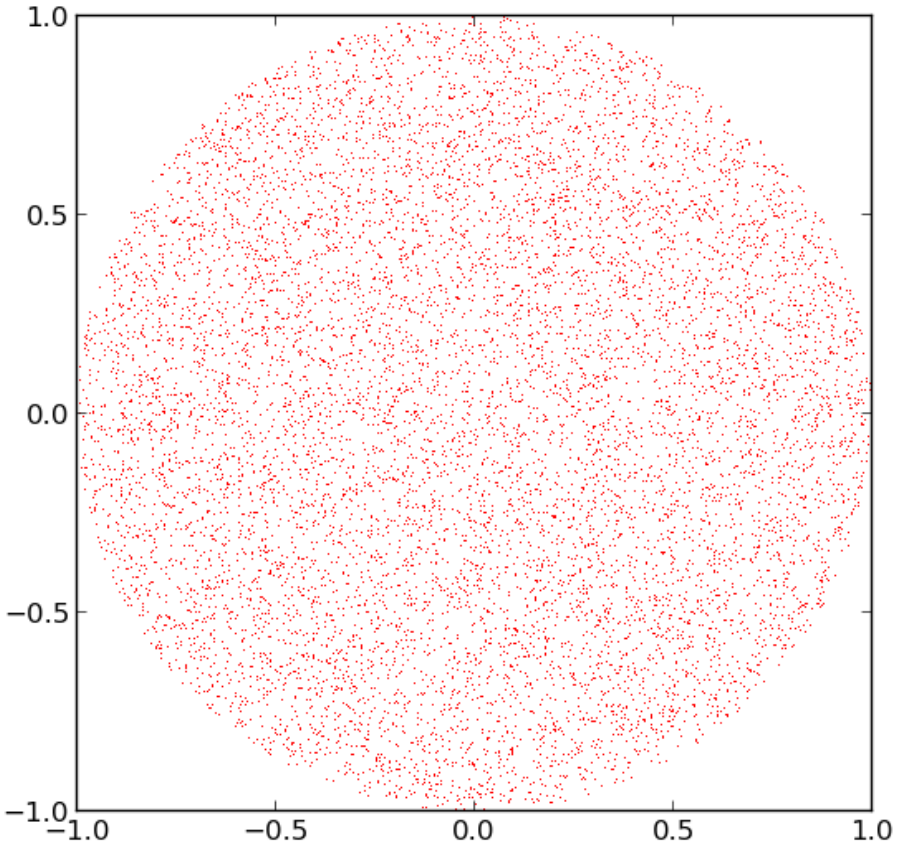
```
import random
random.seed(32)
for i in xrange(16):
    qs = [random.randint(0,40) for j in xrange(100)]
    s,e = trim(qs, 20)
    assert s <= e
    assert np.all(qs[s:e] > 20)
```

Other Random Things

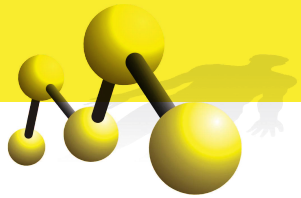


- Random floating point numbers
- Random normally distributed values
- Shuffle arrays
- ...

Random on a circle

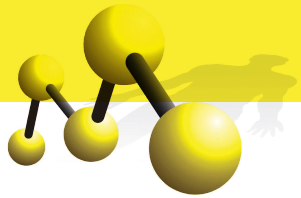


More randomness



- Check out numpy.random
- Check out scipy.stats

Pickle

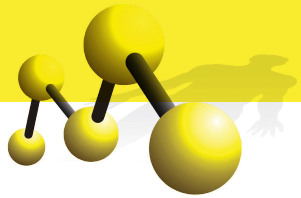


```
import pickle

something = [12, 'hello']

pickle.dump(something, open('myfile.pkl', 'w'))
```

Later

```
import pickle

other = pickle.load(open('myfile.pkl'))
```

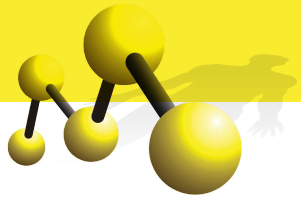
Break



Two minute break before we change the subject

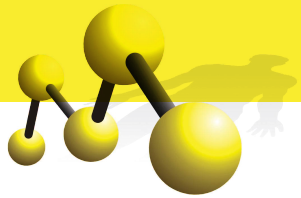
- Talk to your neighbours
- Breath
- Ask questions

Review



Review of course material

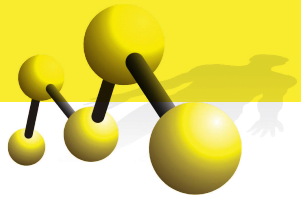
Review



Course Content: Python

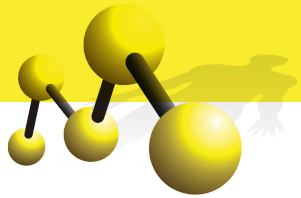
- Basic types: int, float, list, dict, set
- Control flow: for, while, if, elif, else...
- Defining types: class, __init__,...
- Errors (Exceptions): try, except, raise,...
- Modules & Standard Library: import

Course Content: Numeric Representations



Memory & Numeric Representations

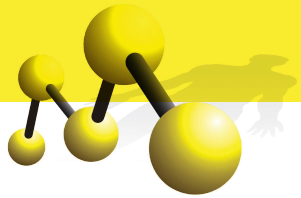
- It's bits all the way down
- Binary representation of signed & unsigned integers
- Floating point numbers
- When handling a lot of data, think of memory usage



Parsing files

- Files are just Bytes (sequence of small numbers)
- It is all in how you interpret them
- There are standard character assignments for text files
- ASCII (English only), Latin-15 (used in Portugal), UTF-8 (usable for everything).

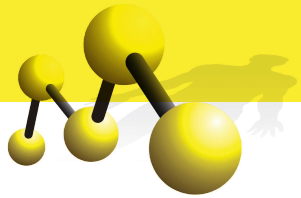
Course Content: Open Source



Open Source

- Free as in beer, free as in speech (gratis/freedom distinction)
- Copyleft vs. liberal licenses
- It is not about price

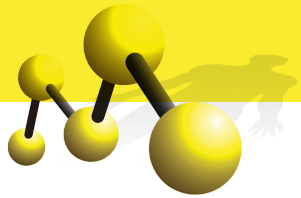
Course Content: Testing



Testing

- Testing is good and you should do it

What Was Not in The Course



Missing

- Some more advanced programming details
- Version Control
- Unix & Shell & Interacting with Other Programmes
- More specific tools