# Python's Odds and Ends (II)
# Error Handling

Luís Pedro Coelho

Programming for Scientists

January 27, 2009

University of Pittsburgh

Carnegie Mellon

## bacteria.py

```
class Bacterium(object):
    ...
...
def simulate(...):
    '''...'''
    ...
...
```

## script.py

```
import bacteria

population = [bacteria.Bacterium(...) ...]
bacteria.simulate(...)
```

# Namespaces

Namespaces are where names live.

# Importing (II)

```
import bacteria
simulate = bacteria.simulate
Bacterium = bacteria.Bacterium
```

# Importing (II)

```
import bacteria
simulate = bacteria.simulate
Bacterium = bacteria.Bacterium

from bacteria import simulate, Bacterium
```

# Importing (III)

```
import bacteria
import bacteria
import bacteria
import bacteria
import bacteria
import bacteria
```

# Import All

```
from bacteria import *
```

# Import As

```
import bacteria
bac = bacteria

bac.simulate(...)
```

bac is another name for bacteria (modules are objects too!)

# Import As

```
import bacteria
bac = bacteria

bac.simulate(...)
```

bac is another name for bacteria (modules are objects too!)

```
import bacteria as bac
```

# Modules & Libraries

```
import math

math.exp(1)
```

# Odds & Ends (II)

```
def enumerate(iterable):
    '''...'''
    i = 0
    for val in iterable:
        yield i,val
        i += 1

buildings = ['Scaiffe','Wean','Mellon']
for i,build in enumerate(buildings):
    print i, build
```

# Zip

```
names = ['Rita','Luis','Sabah']
grades = ['A','B','A']

for g,n in zip(names,grades):
    print 'Student %s had grade %s' % (g,n)
```

# File Reading

```
for line in file('filename.txt'):
    print line
```

# File Objects

```
input = file('myfile.data')
for line in input:
    ...

output = file('myoutput.txt','w')
print >>output, 'Hello World\n'
output.close()
```

# File Objects

```
input = file('myfile.data','r')
for i in xrange(5):
    # First five lines are comments
    input.readline()

sixth = input.readline()
rest = input.readlines()
```

# Errors

Errors happen.

# Errors

- File doesn't exist.
- File is misformatted.
- Some data is not appropriate for the situation.
- You have a bug in your code.
- ...

# Handling Errors

What should a function that reads a file do if the file doesn't exist?
What should the log() function return for negative numbers?

## 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>
```

## Exceptions

- Exceptions are objects.
- Exceptions have type and values.

# Exception Hierarchy

(Nothing here, folks, look at the blackboard)

# Exception Handling

## Exception Handling: Error Handling

```
try:
    <code 1>
    <code 2>
    <code 3>
except IOError, exc:
    print 'I/O Error', exc
except:
    print 'Unspecified error'
finally:
    <code 1>
    <code 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)
True
True

(b)
True
False

(c)
False
Exception

(d)
True
Exception