Python: Modular Code, Handling Files and Objects

Asokan Pichai
Prabhu Ramachandran

Department of Aerospace Engineering
IIT Bombay

Day 1, Session-3, 10, October 2009

Outline

Outline

Problem set 3

As you can guess, idea is to use `for` !

Problem 3.1

Which of the earlier problems is simpler when we use `for` instead of `while` ?

Problem 3.2

Given an empty chessboard and one Bishop placed in any square, say (r, c), generate the list of all squares the Bishop could move to.

Problem 3.3

Given two real numbers `a, b`, and an integer `N`, write a function named `linspace( a, b, N)` that returns an ordered list of `N` points starting with `a` and ending in `b` and equally spaced.
For example, `linspace(0, 5, 11)`, should return,

```
[ 0.0 ,  0.5,  1.0 ,  1.5,  2.0 ,  2.5,
  3.0 ,  3.5,  4.0 ,  4.5,  5.0 ]
```

Problem 3.4a (optional)

Use the `linspace` function and generate a list of N tuples of the form

$[(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_N, f(x_N))]$

for the following functions,

- `f(x) = sin(x)`
- `f(x) = sin(x) + sin(10*x).`

Problem 3.4b (optional)

Using the tuples generated earlier, determine the intervals where the roots of the functions lie.

15 m

Outline

Simple tokenizing and parsing

```
s = """The quick brown fox jumped
        over the lazy dog"""
for word in s.split():
    print word.capitalize()
```

Problem 4.1

Given a string like, “1, 3-7, 12, 15, 18-21”, produce the list

```
[1,3,4,5,6,7,12,15,18,19,20,21]
```

File handling

```
>>> f = open('/path/to/file_name')
>>> data = f.read() # Read entire file.
>>> line = f.readline() # Read one line
>>> f.close() # close the file.
```

Writing files

```
>>> f = open('/path/to/file_name', 'w')
>>> f.write('hello world\n')
>>> f.close()
```

- Everything read or written is a string

Try `file?` for more help

File and for

```
>>> f = open('/path/to/file_name')
>>> for line in f:
...         print line
...
```

Problem 4.2

The given file has lakhs of records in the form:
`RGN;ID;NAME;MARK1;...;MARK5;TOTAL;PFW`
Some entries may be empty. Read the data from this file and print the name of the student with the maximum total marks.

Problem 4.3

For the same data file compute the average marks in different subjects, the student with the maximum mark in each subject and also the standard deviation of the marks. Do this efficiently.

45 m

Outline

Modules

```
>>> sqrt(2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'sqrt' is not defined
>>> import math
>>> math.sqrt(2)
1.4142135623730951
```

Modules

- The `import` keyword "loads" a module
- One can also use:

```
>>> from math import sqrt
>>> from math import *
```

- What is the difference?
- Use the later only in interactive mode

Package hierarchies

```
>>> from os.path import exists
```

Modules: Standard library

- Very powerful, “Batteries included”
- Some standard modules:
  - Math: `math`, `random`
  - Internet access: `urllib2`, `smtplib`
  - System, Command line arguments: `sys`
  - Operating system interface: `os`
  - Regular expressions: `re`
  - Compression: `gzip`, `zipfile`, and `tarfile`
  - And a whole lot more!
- Check out the Python Library reference: `http://docs.python.org/library/`

Modules of special interest

`numpy`	Efficient, powerful numeric arrays
`matplotlib`	Easy, interactive, 2D plotting
`scipy`	statistics, optimization, integration, linear algebra, Fourier transforms, signal and image processing, genetic algorithms, ODE solvers, special functions, and more
Mayavi	Easy, interactive, 3D plotting

Creating your own modules

- Define variables, functions and classes in a file with a `.py` extension
- This file becomes a module!
- Accessible when in the current directory
- Use `cd` in IPython to change directory
- Naming your module

Modules: example

```
# --- arith.py ---
def gcd(a, b):
    if a%b == 0: return b
    return gcd(b, a%b)
def lcm(a, b):
    return a*b/gcd(a, b)
# ------------------
>>> import arith
>>> arith.gcd(26, 65)
13
>>> arith.lcm(26, 65)
130
```

Problem 5.1

Put all the functions you have written so far as part of the problems into one module called `bprim.py` and use this module from IPython.

70 m

Outline

Readability and Consistency

- Readability Counts!-Code is read more often than its written.
- Consistency!
- Know when to be inconsistent.

Code Layout

- Indentation
- Tabs or Spaces??
- Maximum Line Length
- Blank Lines
- Encodings

Whitespaces in Expressions

- When to use extraneous whitespaces??
- When to avoid extra whitespaces??
- Use one statement per line

Comments

- No comments better than contradicting comments
- Block comments
- Inline comments

Docstrings

- When to write docstrings?
- Ending the docstrings
- One liner docstrings

80 m

Outline

Objects in Python

- What is an Object? (Types and classes)
- identity
- type
- method

Why are they useful?

```
for element in (1, 2, 3):
    print element
for key in {'one':1, 'two':2}:
    print key
for char in "123":
    print char
for line in open("myfile.txt"):
    print line
for line in urllib2.urlopen('http://site.com'):
    print line
```

And the winner is . . . OBJECTS!

All objects providing a similar inteface can be used the same way.
Functions (and others) are first-class objects. Can be passed to and returned from functions. 90 m