

Lezione 4

Bioinformatica

Mauro Ceccanti[‡] e Alberto Paoluzzi[†]

[†]Dip. Informatica e Automazione – Università “Roma Tre”
[‡]Dip. Medicina Clinica – Università “La Sapienza”

Lecture 4: Python Overview

Installing Python

- Using Python Interactively
- Running Python Programs
- Setting up Emacs for Python

Basics and Control Flow

- Syntax, Variables and Namespaces

Python Data Structures

- Basic Data Types
- Container Data Structures

Files and Modules

- Import statement
- Documentation



Contents

Lecture 4: Python Overview

Installing Python

- Using Python Interactively
- Running Python Programs
- Setting up Emacs for Python

Basics and Control Flow

- Syntax, Variables and Namespaces

Python Data Structures

- Basic Data Types
- Container Data Structures

Files and Modules

- Import statement
- Documentation

Installing Python

- ▶ If you don't know which version to use, start with Python 2.6.3;
- ▶ more existing third party software is compatible with Python 2 than Python 3 right now.
- ▶ See the main [Documentation page](#).

[Download Python](#)



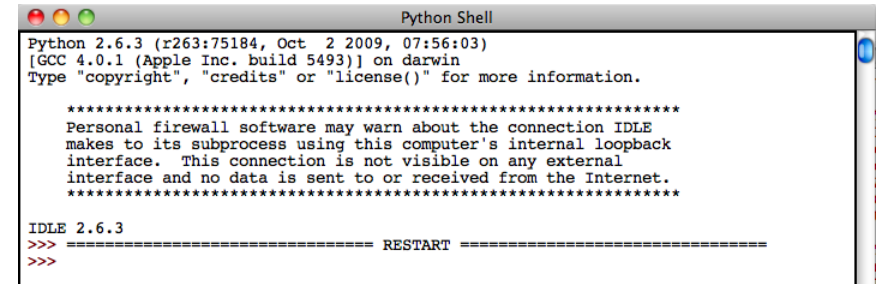
Using Python Interactively

```
baruc3:~> python
Python 2.6.3 (r263:75184, Oct 2 2009, 07:56:03)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license()" for
more information.
>>> 2 + 2
4
>>> 3*(4 + 2)
File "<stdin>", line 1
    3*(4 + 2)
        ^
SyntaxError: invalid syntax
>>> 3 * (4 + 2)
18
>>>
```

Using Python Interactively

Using Idle IDE

Idle is an integrated development environment for Python, which is bundled in each release



```
Python Shell
Python 2.6.3 (r263:75184, Oct 2 2009, 07:56:03)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface. This connection is not visible on any external
interface and no data is sent to or received from the Internet.
*****

IDLE 2.6.3
>>> ===== RESTART =====
>>>
```

- ▶ Multi-window text editor with syntax highlighting, autocompletion, smart indent and other.
- ▶ Python shell with syntax highlighting.
- ▶ Integrated debugger with stepping, persistent breakpoints, and call stack visibility

Running Python Programs

- ▶ Most Python programmers write their programs to a plain file using a text editor.
- ▶ Instead of ending the file with a '.txt' suffix, Python programs usually end with '.py', like 'myprogram.py'.
- ▶ To execute the program file, type the following at a command prompt:

```
> python myprogram.py
```

- ▶ Unix-like systems allow you to add a "bang line" to your program.
- ▶ A "bang line" is a line that tells the computer where to find the interpreter.

```
#!/usr/bin/python
```

- ▶ the execution in this case is direct (give execution rights to the file!)

```
> ./myprogram.py
```

Running Python Programs

```
#!/usr/bin/python
print "ciao"
```

1. move to the file directory `cd dirname`
2. verify file rights `ls -l filename`
3. add execution right `chmod +x filename`
4. program execution `./filename`

```
host:~ pao> cd test
host:test pao> ls -l ciao.py
-rw-r--r--@ 1 paoluzzi staff 30 Oct 22 11:24 ciao.py
host:test pao> chmod +x ciao.py
host:test pao> ls -l ciao.py
-rwxr-xr-x@ 1 paoluzzi staff 30 Oct 22 11:24 ciao.py
host:test pao> ciao.py
host:test pao> ./ciao.py
ciao
host:test pao>
```

Contents

Lecture 4: Python Overview

Installing Python

- Using Python Interactively
- Running Python Programs
- Setting up Emacs for Python

Basics and Control Flow

- Syntax, Variables and Namespaces

Python Data Structures

- Basic Data Types
- Container Data Structures

Files and Modules

- Import statement
- Documentation

Basics and Control Flow

Learning to program in Python

See also: [Discover Python, Part 5: Programming in Python](#)



Syntax, Variables and Namespaces

Python - Variable Types

Contents

Lecture 4: Python Overview

Installing Python

- Using Python Interactively
- Running Python Programs
- Setting up Emacs for Python

Basics and Control Flow

- Syntax, Variables and Namespaces

Python Data Structures

- Basic Data Types
- Container Data Structures

Files and Modules

- Import statement
- Documentation



Python Data Structures

- ▶ Lists
- ▶ Tuples
- ▶ Dictionaries
- ▶ Sets

Data Structures

Contents

Lecture 4: Python Overview

Installing Python

- Using Python Interactively
- Running Python Programs
- Setting up Emacs for Python

Basics and Control Flow

- Syntax, Variables and Namespaces

Python Data Structures

- Basic Data Types
- Container Data Structures

Files and Modules

- Import statement
- Documentation

Container Data Structures

- ▶ In computer science, a container is a class, a data structure, or an abstract data type (ADT) whose instances are collections of other objects.
- ▶ In other words; They are used to store objects in an organized way following specific access rules.
- ▶ Generally, container classes are expected to implement methods to do the following:
 - ▶ create a new empty container (constructor),
 - ▶ report the number of objects it stores (size),
 - ▶ delete all the objects in the container (clear),
 - ▶ insert new objects into the container,
 - ▶ remove objects from it,
 - ▶ provide access to the stored objects.

Modules

From Python tutorial, see: [6. Modules](#)

See also: [Importing Python Modules](#)

From Python tutorial, see: [7.2. Reading and Writing Files](#)

Documentation

[Source Documentation and Comments](#)

Import statement

See within the files: `wireframe.py`, `pdb.py`, and `basic.py`

```
"""  
To generate the wireframe of a protein.  
Usage:  
> python wireframe.py protein.pdb  
"""  
from pdb import *
```

```
import sys  
import datetime  
from basic import *  
import FL06
```

```
"""  
Basic data and operations with amino acids and  
biomolecules.  
"""  
from numpy import array, reshape, transpose
```

