Métodos Computacionales

FISI 2028

========== Python==========

Python

Universidad de los Andes
Departamento de Física

2015

Python

Where is it used?

Web Development, Yahoo Maps, Yahoo Groups, Google, Zope Corporation, Ultraseek, Linux Weekly News, ElasticHosts Cloud Servers, Mojam.com, hunch, Shopzilla, Movieplayer.it, Multiplayer.it, Web Therapy, Battlefield 2, Crystal Space, Star Trek Bridge Commander, The Temple of Elemental Evil and Vampire: The Masquerade: Bloodlines, Civilization 4, QuArK (Quake Army Knife), Industrial Light & Magic, Walt Disney Feature Animation, HKS, Inc. (ABAQUS/CAE), RoboFog, Caligari Corporation, Blender 3D, Jasc Software, Paint Shop Pro, Altis Investment Management, ABN AMRO Bank, Treasury Systems, Bellco Credit Union, Journyx Timesheet and Resource Management Software, National Weather Service, Applied Maths, Biosoft, The National Research Council of Canada, Los Alamos National Laboratory (LANL) Theoretical Physics Division, AlphaGene, Inc., LLNL, NASA, Swedish Meteorological and Hydrological Institute (SMHI), Environmental Systems Research Institute (ESRI), Nmag Computational Micromagnetics, Objexx Engineering, Ciranova, Productivity Design Tools, PDTi's SpectaReg product is an eXtensible Memory-Mapped Register generator., Object Domain, Pardus, Red Hat, SGI, Inc., MCI Worldcom, Nokia, University of California, Irvine, Delft University of Technology, Faculty of Aerospace Engineering, Delft, Netherlands, Smeal College of Business, The Pennsylvania State University, New Zealand Digital Library, IT Certification Exam preparation, SchoolTool, Raven Bear Systems Corporation, Thawte Consulting, Advanced Management Solutions Inc., IBM, RealNetworks, dSPACE, Escom, The Tiny Company, Nexedi, WuBook

~1~

Readability counts.
Now is better than never.
Flat is better than nested.
Unless explicitly silenced.
Sparse is better than dense.
Beautiful is better than ugly.
Simple is better than complex.
Explicit is better than implicit.
Errors should never pass silently.
Although practicality beats purity.
Complex is better than complicated.
Although never is often better than *right* now.
Special cases aren't special enough to break the rules.
In the face of ambiguity, refuse the temptation to guess.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.

>> highly extensible
>> nice to read
>> object-oriented
>> structured

The Zen of Python,
By Tim Peters

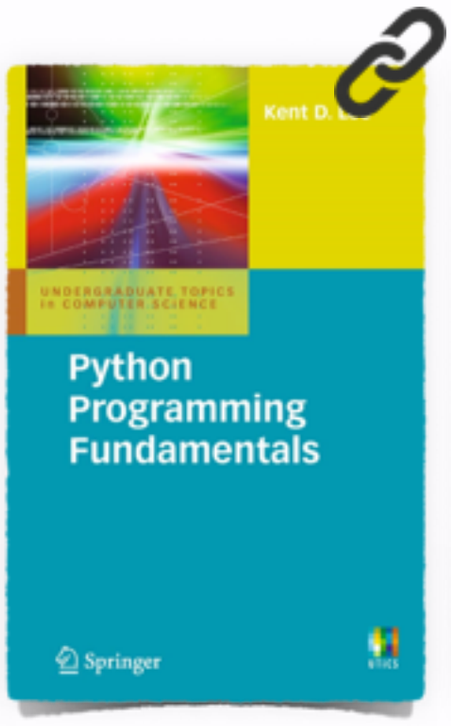
Guido van Rossum

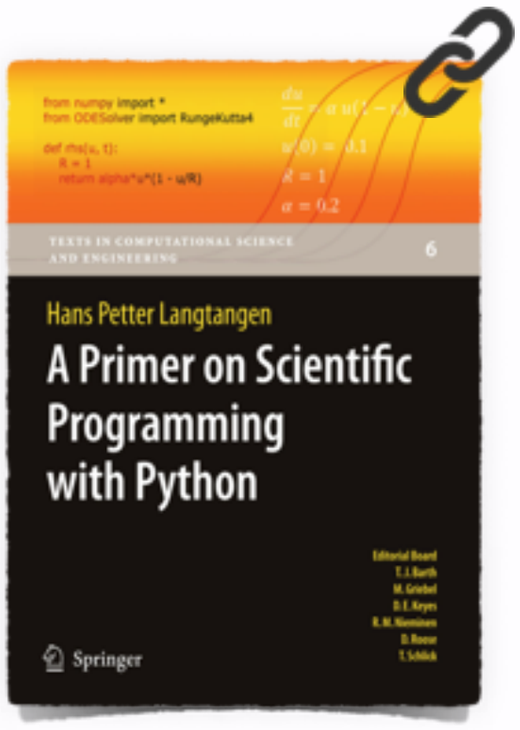
¿Cómo instalar Python?

Anaconda

http://continuum.io/downloads

Referencias

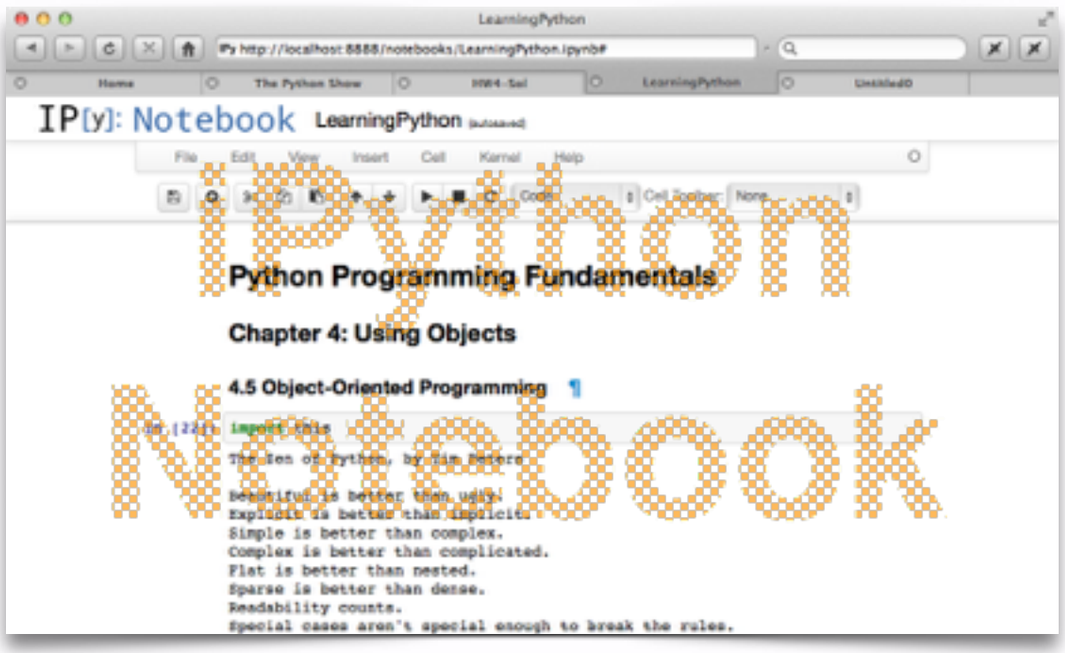


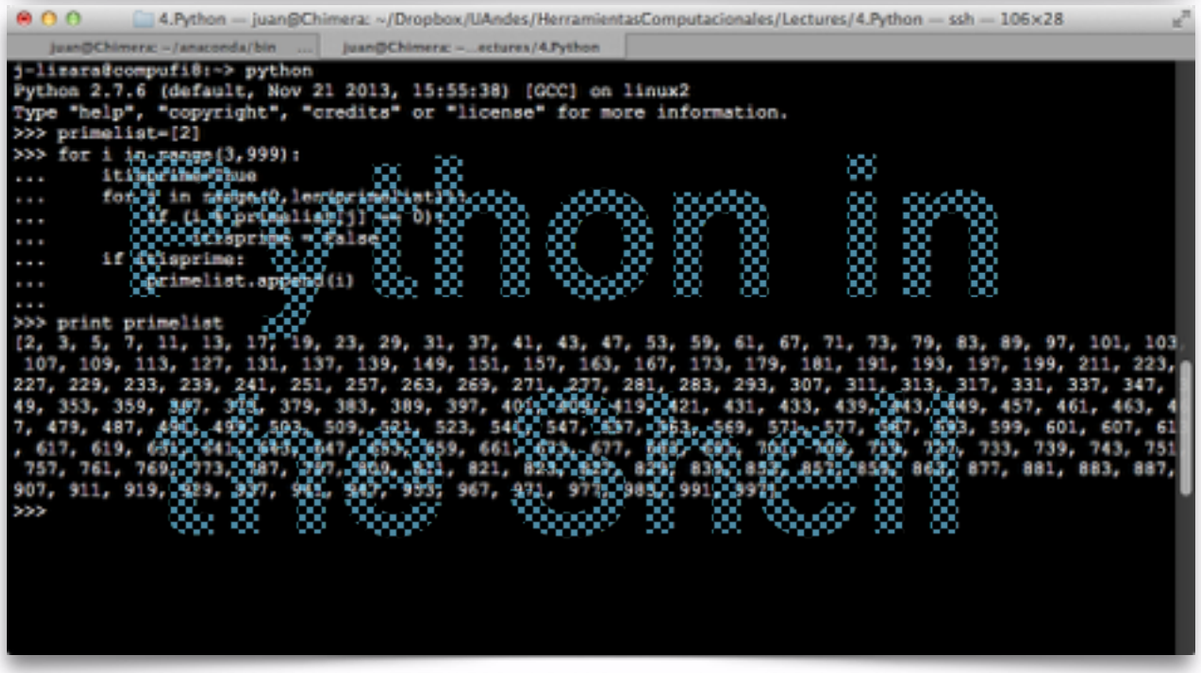


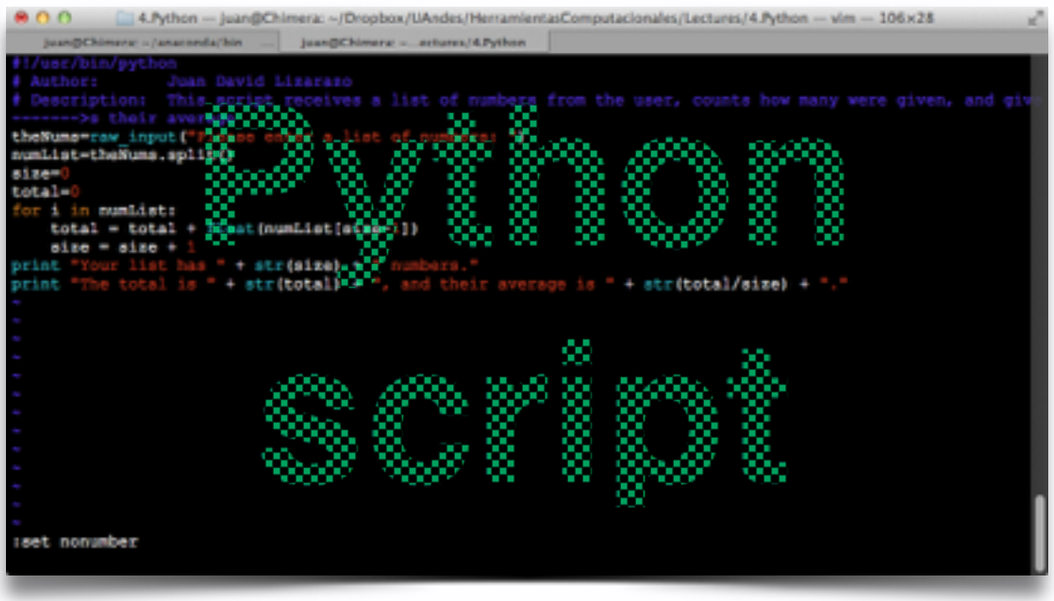
IDE

PyCharm

4 Python Vessels








```
j-lizara@compufi8:~> python
Python 2.7.6 (default, Nov 21 2013, 15:55:38) [GCC] on linux2
Type "help", "copyright", "credits" or "license" for more info.
>>>
```

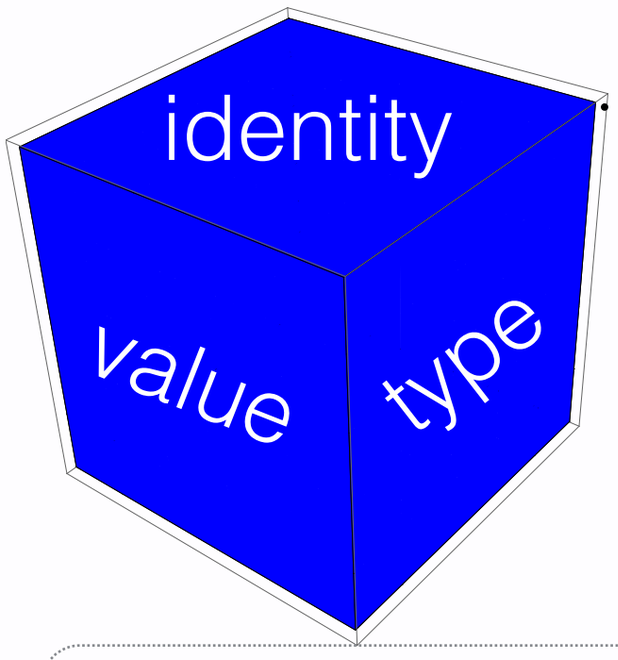
```
#!/bin/bash
```

bash script

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

Python script

Todo es un objeto



identifier

assignment

any expression
which can be evaluated

```
<identifier> = <expression>
```

```
x = 10
```

tipos de objetos:
int, float, bool, list, …

> Objects are Python's abstraction for data. All data in a Python program is represented by objects or by relations between objects. (In a sense, and in conformance to Von Neumann's model of a "stored program computer," code is also represented by objects.)
>
> Every object has an identity, a type and a value.

> [...] An object's type determines the operations that the object supports (e.g., "does it have a length?") and also defines the possible values for objects of that type.

Taken from the Python data model reference

Operaciones Aritméticas

Operación	Operador	Comentarios
Addition	x + y	x and y may be floats or ints
Subtraction	x - y	x and y may be floats or ints
Multiplication	x * y	x and y may be floats or ints
Division	x / y	If at least one of x and y is a float, then the result is also a float, be cautious when dividing integers.
Floor Division	x // y	x and y may be floats or ints. The result is the first integer less than or equal to the quotient
Remainder or Modulo	x % y	x and y must be ints. This is the remainder of diving x by y.
Exponentiation	x ** y	x and y can be floats or ints. This is the result of raising x to ty yth power.
Float conversion	float(x)	Convert the numeric value of x to a float.
Integer Conversion	int(x)	Convert the numeric value of x to an int. The decimal portion is truncated not rounded.
Absolute Value	abs(x)	Gives the absolute value of x.
Round	round(x)	Rounds the float, x, to the nearest whole number. The result is alwasy an int.

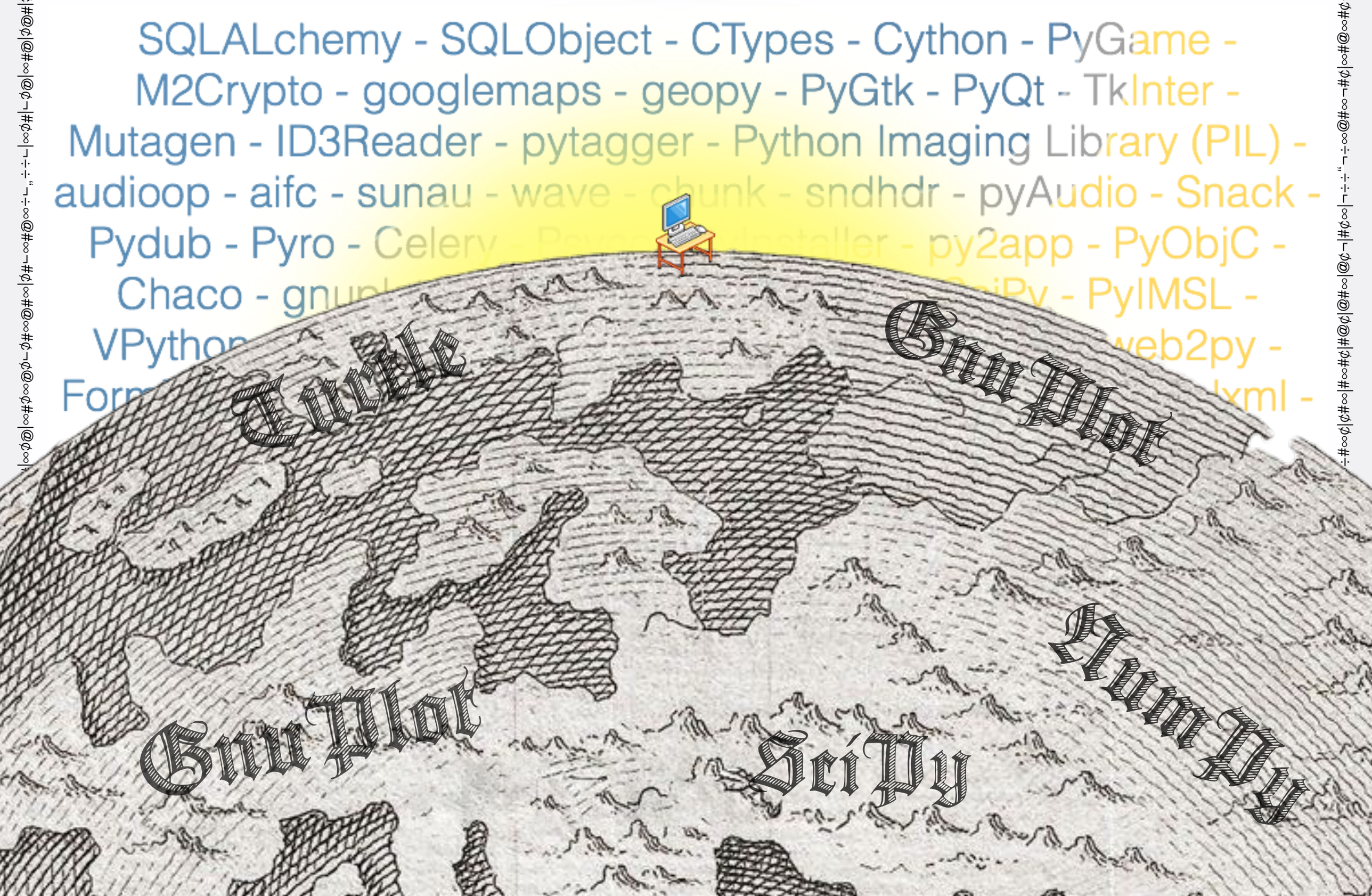
(Tabla tomada del libro de Lee)

Operaciones con Cadenas

Operación	Operador	Comentarios
Indexing	x[x]	Yields the xth character of the string s. The index is zero based, so s[0] is the first character.
Concatenation	s + t	Yield the juxtaposition of the strings s and t.
Length	len(s)	Yields the number of characters in s.
Ordinal Value	ord(x)	Yields the ordinal value of a character c. The ordinal value is the ASCII code of the character.
Character Value	chr(x)	Yields the character that corresponds to the ASCII value of x.
String Conversion	str(x)	Yields the string representation of the value of x. The value of x may be an int, float, or other type of value.
Integer Conversion	int(s)	Yields an integer value contained in the string s. If s does not contain an integer an error will occur.
Float Conversion	float(s)	Yields the float value contained in the string s. If s does not contain a float an error will occur.

(Tabla tomada del libro de Lee)

Augmenting Python



¿Cómo solicitar información al usuario?

```
>>> x=raw_input("Please enter x=")
Please enter x = 
```

raw_input recibe strings, si se espera un número y quieren hacerse operaciones aritméticas, debe entonces convertirse en un objeto de tipo numérico.

```
>>> x=float(raw_input("x="))
x=10.2
>>> print x + 2
12.2
```

Si se espera que el usuario entregue una lista (con elementos separados por espacios en blanco) usar raw_input seguido the el método split aplicado sobre la cadena recibida, lo cual produce una lista que contiene sus elementos.

```
>>> theList=raw_input("Please enter a list of things: ")
Please enter a list of things: gato 2 3 4
>>> theSlist = theList.split()
>>> print theSlist
['gato', '2', '3', '4']
```

Expresiones lógicas, if & while

```
a == b  #evalúa si a es igual a b
a != b  #evalúa si a es distinto de b
a < b   #evalúa si a es menor que b
a <= b  #evalúa si a es menor o igual a b
```

and or

if

```
if condition:
    thing_to_do1
    thing_to_do2
```

if … else

```
if condition:
    thing_to_do1
    thing_to_do2
else:
    other_thing1
    other_thing2
```

¡La indentación es esencial!

while

```
while condition:
    thing_to_do1
    thing_to_do2
```

```
# This code prints all ordered pairs of
# integers with the integers ranging
# from 1 to 10. Look with caution at the
# usage of indentation
j = 1
while j <= 10:
    i = 1
    while i <= 10:
        print "{" + str(j) + "," + \
         str(i) + "}",
        i += 1
    j += 1
```

for

```
for i in range(1,10):
    thesquare = i**2
    print str(i)+"^2="+str(thesquare)
```

```
1^2=1
2^2=4
3^2=9
4^2=16
5^2=25
6^2=36
7^2=49
8^2=64
9^2=81
```

```
for x in range(10):
    y = 2*x
    if x == 3:
        break
    print y
```

```
0
2
4
```

Use **break** to break out of *for* or *while* loops

```
emptys=""
for s in "Juan David Lizarazo":
    emptys = s + emptys
print emptys
```

```
ozaraziL divaD nauJ
```

It is possible to iterate over strings

```
for i in iterableObj:
    todo1
    todo2
    todo3
    todo4
```

Iterable Objects

strings,
lists, arrays

range(n) gives a list of integers from 0 to n-1

range(a,b) gives a list of integers from a to b-1

Listas

```
>>> lista = ["uno", "dos", "tres", 2]
```

Los elementos de una lista se dan entre corchetes cuadrados y sus elementos pueden ser de diferente tipo.

```
>>> print lista[0]
uno
>>> lista[-1] += 1
print lista
['uno', 'dos', 'tres', 3]
>>> lista.append(4)
print lista
['uno', 'dos', 'tres', 3, 4]
```

El n-ésimo elemento de la lista se invoca ingresando lista[n-1].
Se pueden usar índices negativos para elegir elementos contando desde el último elemento, por ejemplo lista[-1] hace referencia al último elemento.

Para añadir elementos a una lista se invoca el método append sobre la misma, por ejemplo **lista.append(4)** toma una lista y le añade al final el número 4.

Otros métodos que pueden invocarse sobre una lista: **pop** (quitar el último elemento), **reverse** (invertir el orden), **extend** (como append pero con listas).

lista[start:stop:step] toma los elementos de la lista comenzando en el índice *start* terminando en el índice *stop-1* en incrementos *step,* por ejemplo range(100)[10:51:2] produce una lista con todos los números pares entre 10 y 50.

list comprehension & dictionaries

```
# List comprehension
x = []
for i in range(10):
    x.append(i**2)
# Is equivalent to
x = [i**2 for i in range(10) ]
```

Los diccionarios son una especie de lista con índices (keys) arbitrarios.

```
# Build a dictionary
sol={}
sol["masa"]=1.9891E30
sol["radio"]=6.955E8
for key in sol.keys():
    print(key,sol[key])
```

iPython

IP[y]: Notebook The Show of Python 2 (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

The Show of Python 2

List Comprehension

```
In [ ]: theMagicNum = int(raw_input("¿Cuál es su número mágico? "))
        theMagicNums = [i * theMagicNum for i in range(1,1000/theMagicNum + 1)]
        print "Los múltiplos de " + str(theMagicNum) + " menores o iguales a 1000 s
        print theMagicNums
        print ", y su suma es " + str(sum(theMagicNums)) + "."
```

Funciones

Simplest example

```
In [4]: def duplicator(x):
            y = x*2
            return y
```

```
In [5]: [ duplicator(i) for i in range(10)]
```

arrays

4.2.2 Basics of Numerical Python Arrays

An *array* object can be viewed as a variant of a list, but with the following assumptions and features:

- All elements must be of the same type, preferably integer, real, or complex numbers, for efficient numerical computing and storage.
- The number of elements must be known[5] when the array is created.
- Arrays are not part of standard Python[6] – one needs an additional package called *Numerical Python*, often abbreviated as NumPy. The Python name of the package, to be used in `import` statements, is `numpy`.
- With `numpy`, a wide range of mathematical operations can be done directly on complete arrays, thereby removing the need for loops over array elements. This is commonly called *vectorization* and may cause a dramatic speed-up of Python programs. Vectorization makes use of the vector computing concepts from Chapter 4.1.3.
- Arrays with one index are often called vectors. Arrays with two indices are used as an efficient data structure for tables, instead of lists of lists. Arrays can also have three or more indices.

The fundamental import statement to get access to Numerical Python array functionality reads

```
from numpy import *
```

```
from numpy import *
```

convert a list to an array
`arr=array(list)`
create an array with n zeroes
`arr=zeroes(n)`
create an array with n floats from p to q
`arr=linspace(p,q,n)`

if f(x) is a scalar function, f(array) is an array with the function applied to each element, this is called **vectorization**

```
import numpy as np
arr = np.array([1,1,1])
```

`from module import *`
merges names
`import module as alias`
makes creates a namespace of the sort alias.object

funciones

```
def funcname(arg1,arg2,…):
    statement1
    …
    statementN
    return something
```

```
def duplicator(x):
    y = 2*x
    return y
[ duplicator(i) for i in range(10)]
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

```
def thetotal(*items):
    tot=0
    for i in items:
        tot+=i
    return tot
print thetotal(1,2,10,-1)
print thetotal(*[1,2,10,-1])
12
12
```

a function with a variable number of arguments

LOCAL

GLOBAL

BUILT-IN

~Python identifier's order of precedence~

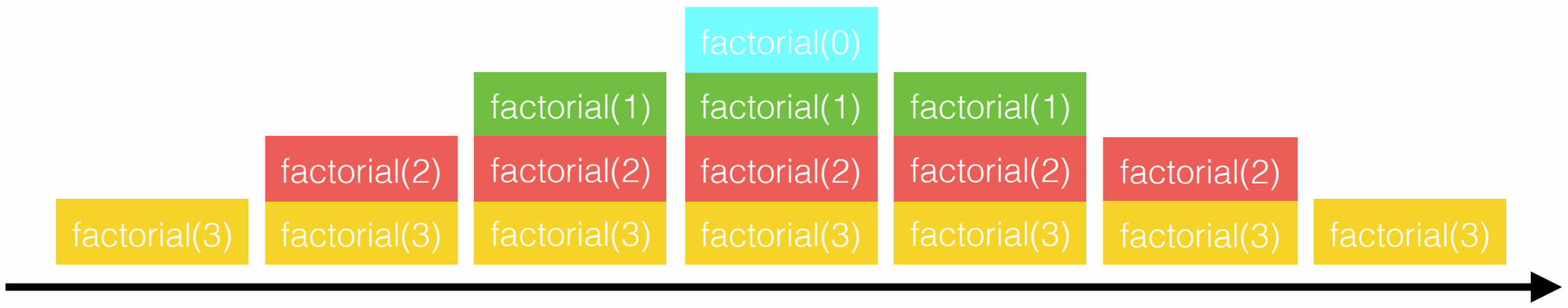
Usar **del** para eliminar la definición de una variable

funciones predeterminadas

abs()	divmod()	input()	open()	staticmethod()
all()	enumerate()	int()	ord()	str()
any()	eval()	isinstance()	pow()	sum()
basestring()	execfile()	issubclass()	print()	super()
bin()	file()	iter()	property()	tuple()
bool()	filter()	len()	range()	type()
bytearray()	float()	list()	raw_input()	unichr()
callable()	format()	locals()	reduce()	unicode()
chr()	frozenset()	long()	reload()	vars()
classmethod()	getattr()	map()	repr()	xrange()
cmp()	globals()	max()	reversed()	zip()
compile()	hasattr()	memoryview()	round()	__import__()
complex()	hash()	min()	set()	apply()
delattr()	help()	next()	setattr()	buffer()
dict()	hex()	object()	slice()	coerce()
dir()	id()	oct()	sorted()	intern()

Funciones recursivas

```
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n-1)
print(factorial(3))
6
```



Attributions

~1~

Today's latte, Python again!

by Yuko Honda.

~modified~