Curso Aprender programación usando Python

Sebastián Bassi (sbassi@genesdigitales.com)

¿Qué es un programa?

Un programa es un conjunto de instrucciones diseñadas para ordenar a la computadora a hacer algo.

Es similar a una receta de cocina, que consiste en una lista de ingredientes e instrucciones paso a paso donde se usan dichos ingredientes.

Primer programa

Código

```
seq1 = 'Hola'
seq2 = ' mundo!'
total = seq1 + seq2
print total
```

Resultado

```
Hola mundo!
```

Ejemplo bajo y alto nivel

Bajo nivel (código máquina x86)

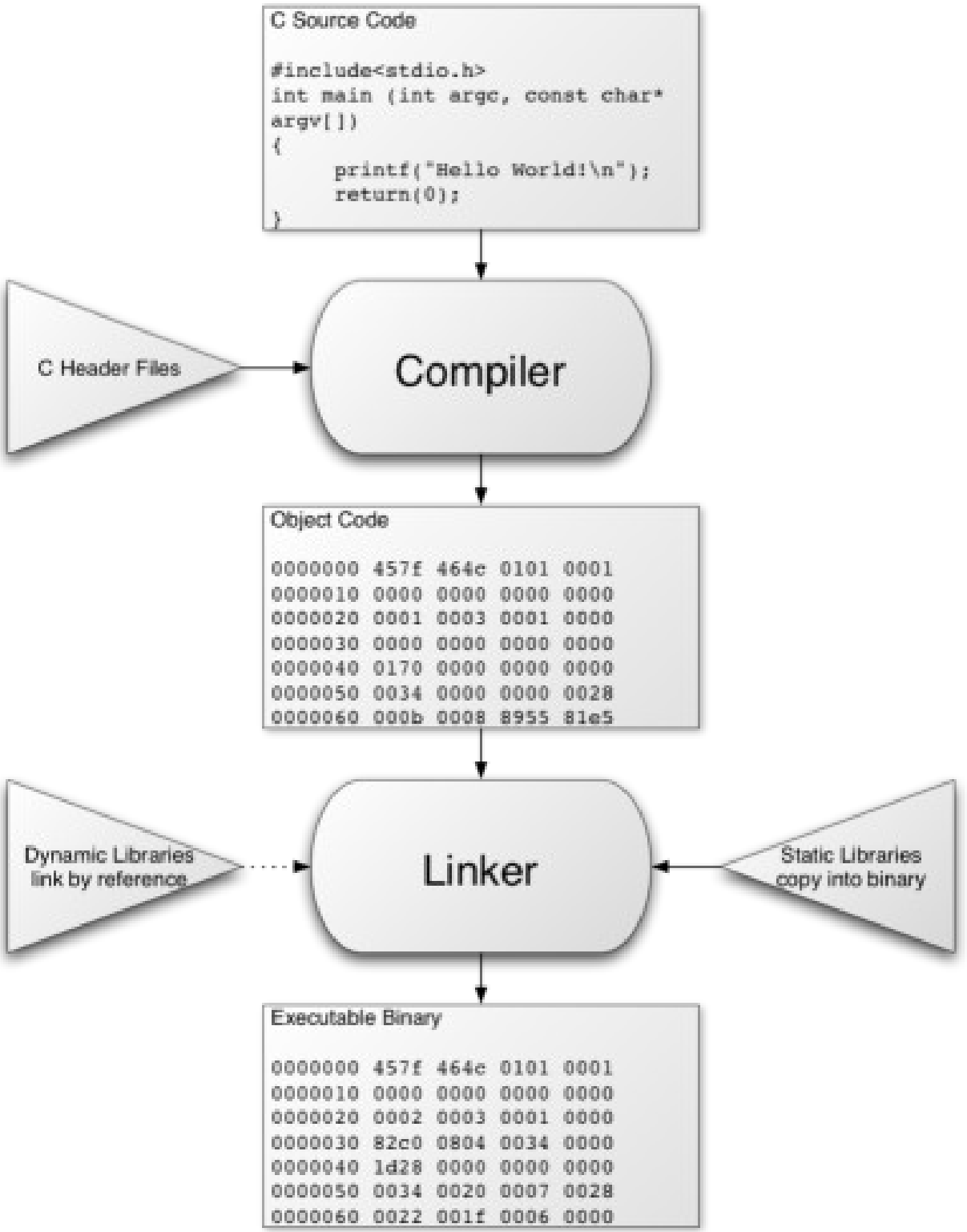
```
8B542408 83FA0077 06B80000 0000C383
FA027706 B8010000 00C353BB 01000000
B9010000 008D0419 83FA0376 078BD98B
C84AEBF1 5BC3
```

Alto nivel (Python)

```
def fib(n):
    a, b = 0, 1
    for i in range(n):
        a, b = b, a + b
    return a
```

Compilación

"Traducción" desde el código fuente a instrucciones "ejecutables"



Consecuencias de la compilación

- Tiempo de compilación
- Aceleración en la ejecución del software
- Software dependiente de una plataforma

Paradigmas de programación

- Procedural / Estructurada: C, Pascal, Perl.
- Orientada a Objetos: C++, Java.
- Lógico: Prolog, Lisp.

Programación procedural

Los programas tienen rutinas o funciones con los pasos a seguir.

Beneficios:

- Estructurar el código en bloques para reutilizarlos.
- Seguimiento de la lógica del programa (sin saltos a posiciones arbitrarias, aka “go to”).

POO (OOP)

Se usan objetos para diseñar los programas. Los objetos son estructuras de datos que tienen propiedades (caracteristicas) y métodos (acciones) que le son propios.

Caracteristicas:

- Encapsulación
- Abstracción de datos
- Polimorfismo
- Herencia

Python: Especificación e implementación

Especificación: Definición de caracteristicas del lenguaje

Implementación: Programa que cumple con dicha especificación. Ej.: CPython, IronPython, Jython

Características de Python

- Fácil de aprender y de programar
- Fácil de leer (similar a pseudocódigo)
- Interpretado (Rápido para programar)
- Datos de alto nivel (listas, diccionarios, sets, etc)
- Libre y gratuito
- Multiplataforma (Win, Linux y Mac)
- Pilas incluidas
- Cantidad de bibliotecas con funciones extras
- Comunidad

Leer archivo y cargarlo en array

VB

```
Dim i, j, Array_Used As Integer
Dim MyArray() As String
Dim InBuffer, Temp As String
Array_Used = 0
ReDim MyArray(50)
'open a text file here . . .
Do While Not EOF(file_no)
    Line Input #file_no, MyArray(Array_Used)
    Array_Used = Array_Used + 1
    If Array_Used = UBound(MyArray) Then
        ReDim Preserve MyArray(UBound(MyArray) + 50)
    End If
Loop
'simple bubble sort
For i = Array_Used - 1 To 0 Step -1
    For j = 1 To i
        If MyArray(j - 1) > MyArray(j) Then
            'swap
            Temp = MyArray(j - 1)
            MyArray(j - 1) = MyArray(j)
            MyArray(j) = Temp
        End If
    Next
Next
```

Leer archivo y cargarlo en lista

Python

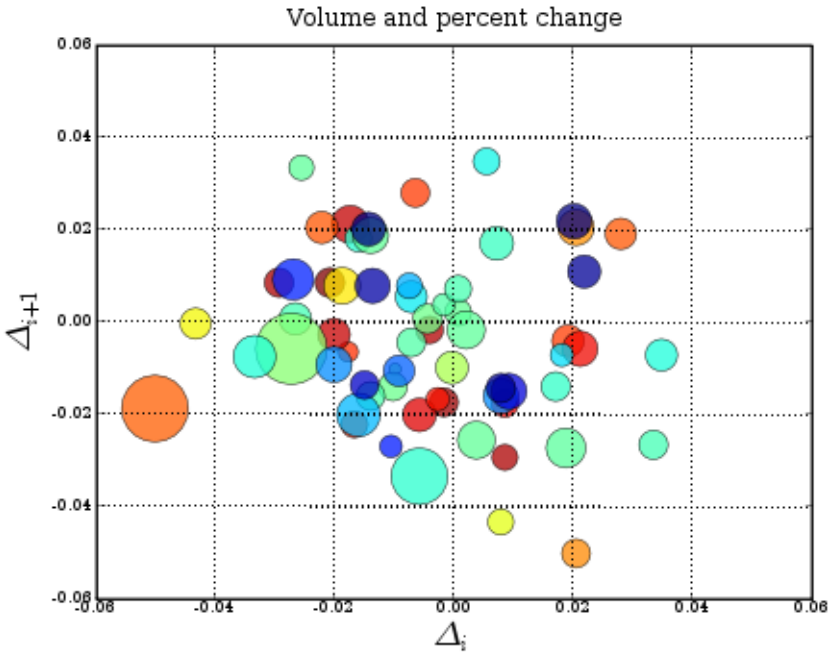
```
# Abrir un archivo de texto . . .
file_object = open(FILENAME)

# Leer todas las lineas del texto en una lista (similar a un array)
lista = file_object.readlines()

# Ordenar la lista
lista.sort()
```

Usos de Python

Procesamiento de datos:

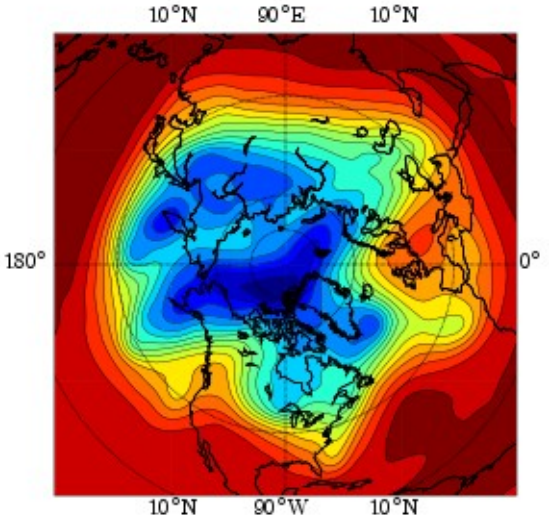
- Conversión de datos de formato arbitrario (html, txt, etc) a csv, para usarse en Excel u otros programas.
- Soporte para lectura y escritura de csv
- Lectura y escritura de XLS (módulos xlrd, xlwt)
- Lectura y escritura de XML (módulos xml.dom, xml.etree.ElementTree, xml.sax, xml.dom)

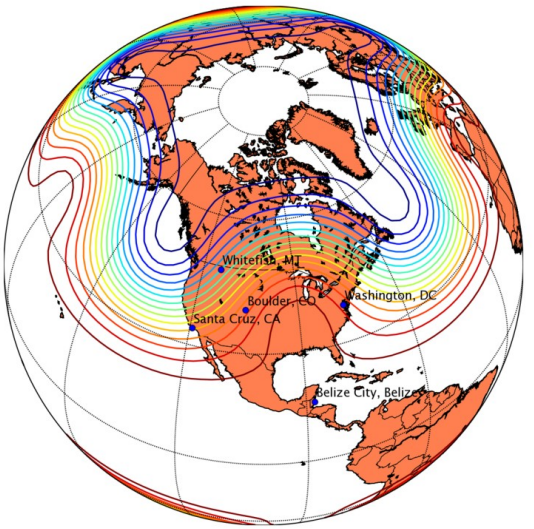
Usos de Python: Gráficos




```
from pylab import *
from data_helper import get_daily_data
intc, msft = get_daily_data()
delta1 = diff(intc.open)/intc.open[0]
# size in points ^2
volume = (15*intc.volume[:-2]/intc.volume[0])**2
close = 0.003*intc.close[:-2]/0.003*intc.open[:-2]
scatter(delta1[:-1], delta1[1:], c=close, s=volume,
alpha=0.75)
ticks = arange(-0.06, 0.061, 0.02)
xticks(ticks)
yticks(ticks)
xlabel(r'$\Delta_i$', fontsize=20)
ylabel(r'$\Delta_{i+1}$', fontsize=20)
title('Volume and percent change')
grid(True)
show()
```

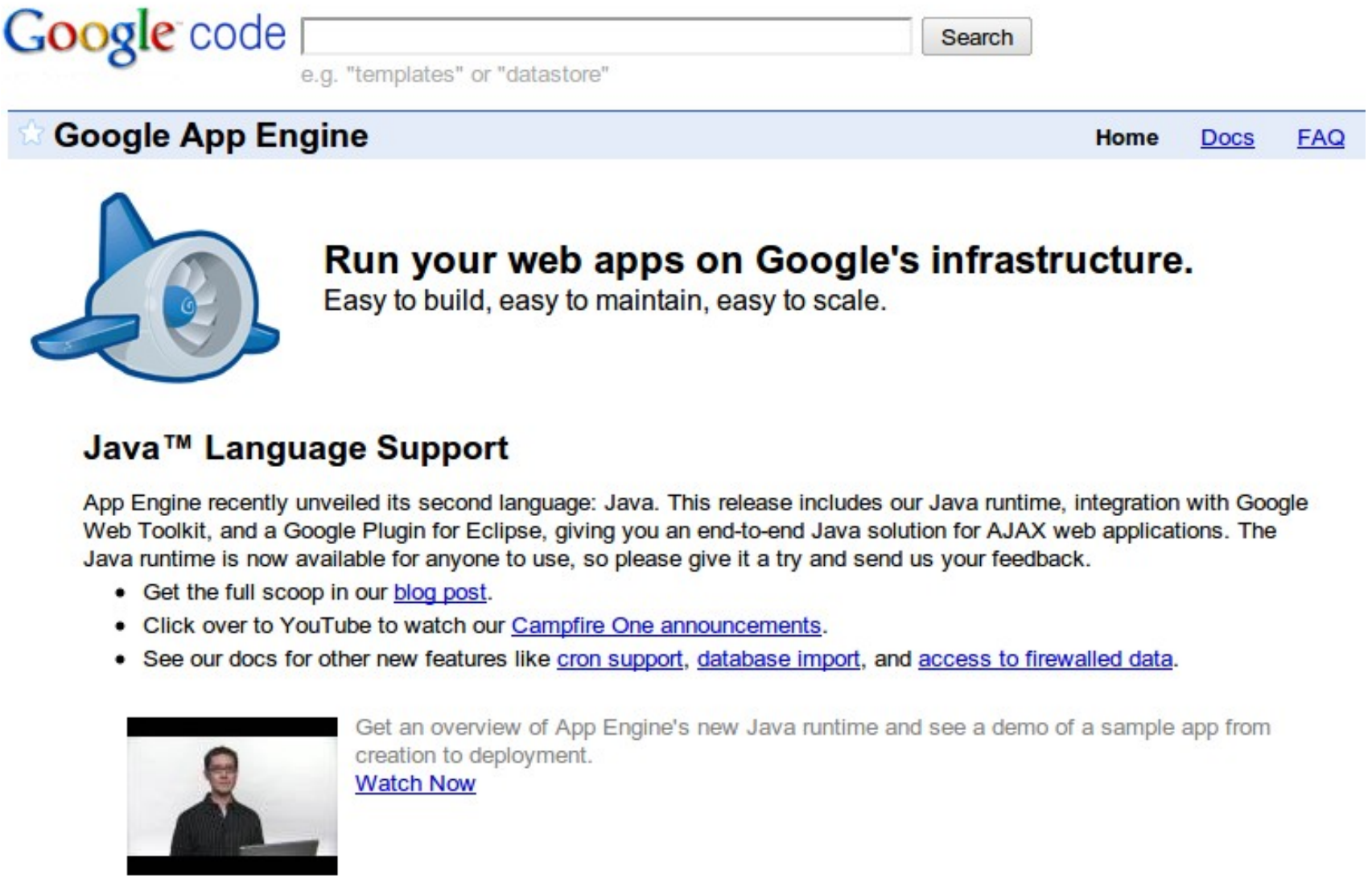
Usos de Python: Gráficos





Usos de Python: Desarrollo web

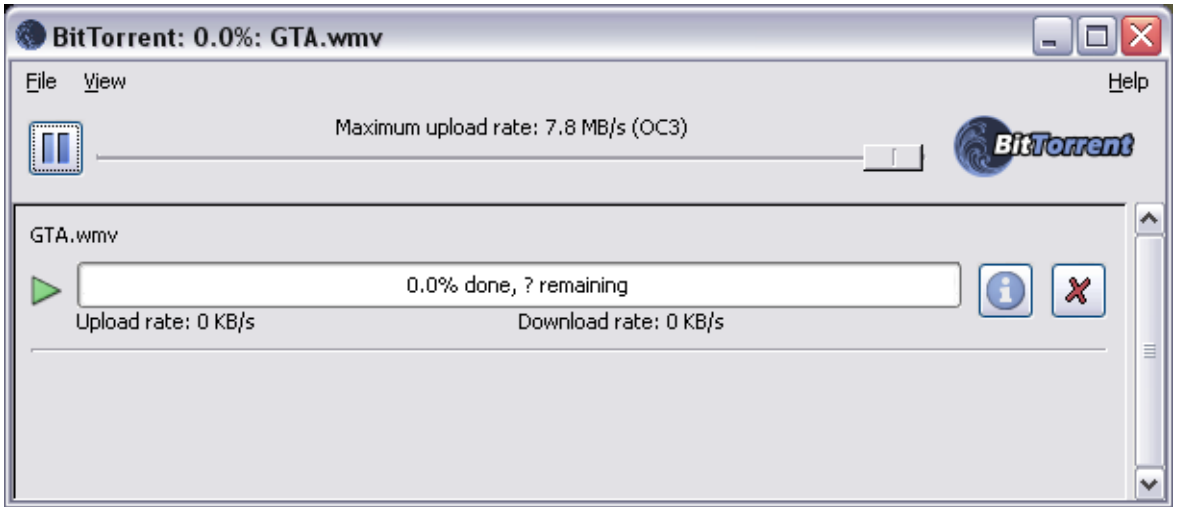
Google App Engine

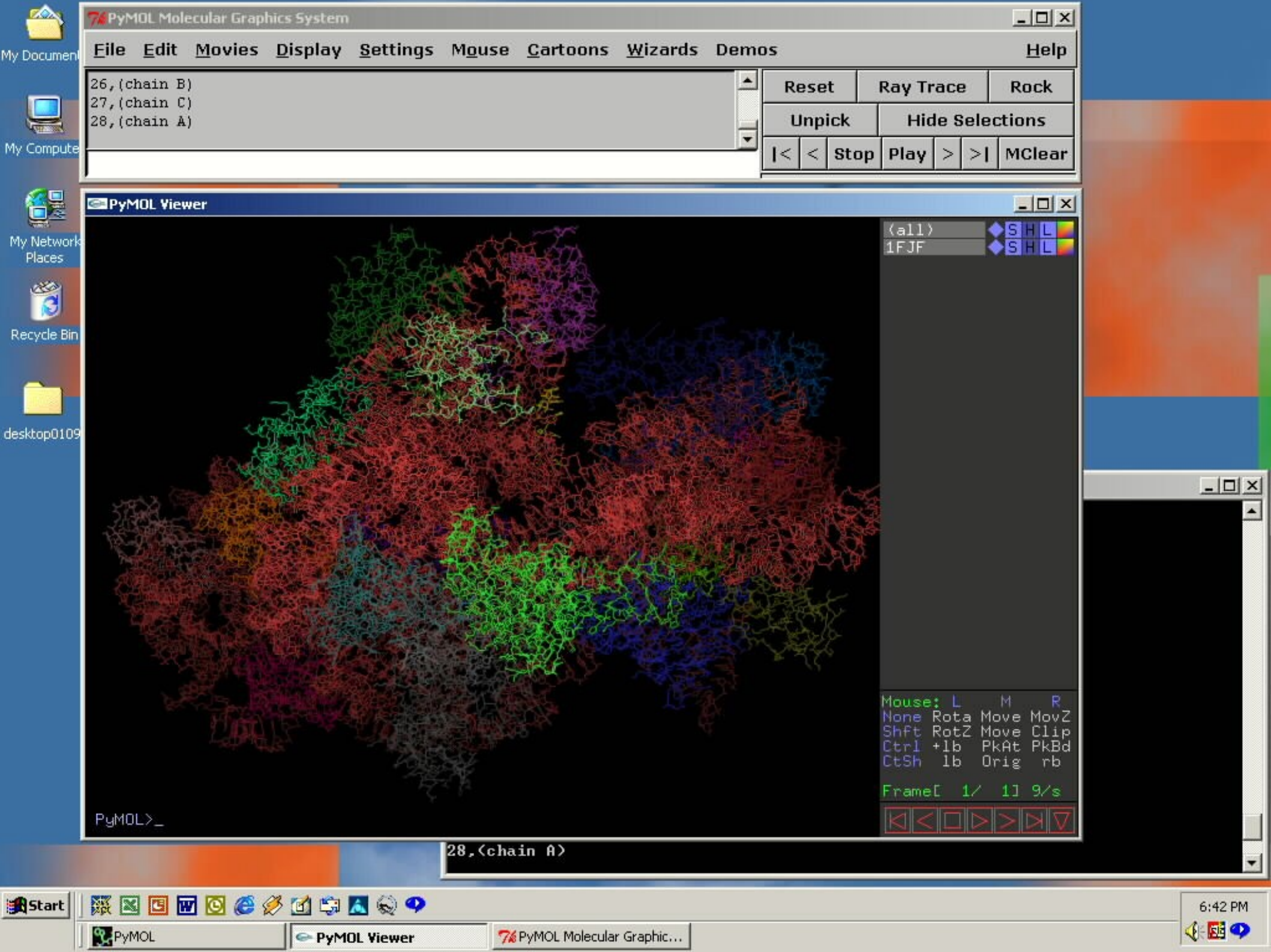


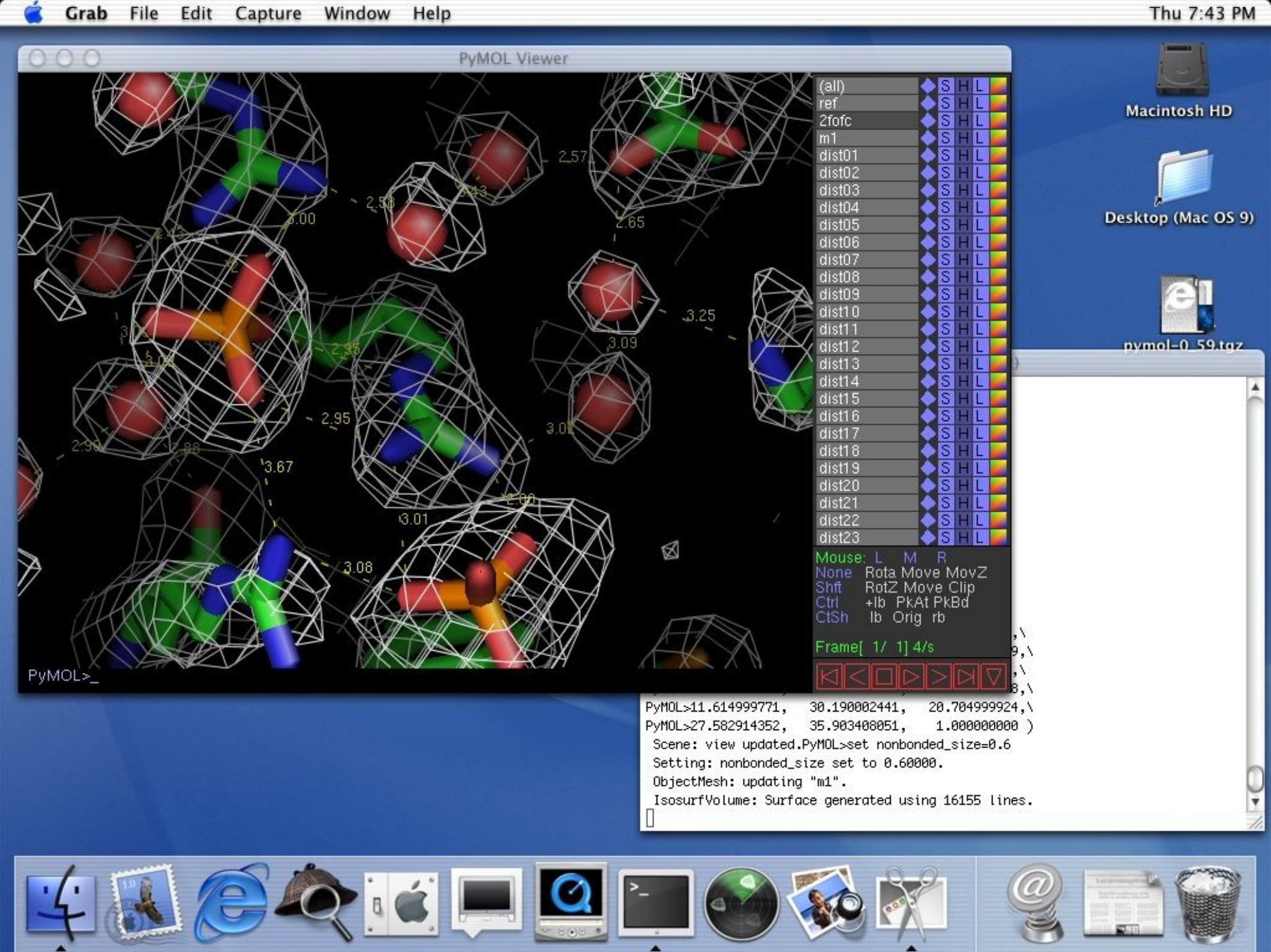
Usos de Python

Acceso a servicios de Internet:

- Traer contenido de páginas web (para traer páginas de manera repetitiva).
- Acceso a POP3, IMAP4 FTP, etc.
- Lee varios formatos mailbox
- Lectura HTML (HTMLparser).
- Aplicación de internet: Bittorrent




PyMOL Molecular Graphics System
File Edit Movies Display Settings Mouse Cartoons Wizards Demos Help
26,(chain B)
27,(chain C)
28,(chain A)
Reset Ray Trace Rock
Unpick Hide Selections
|< < Stop Play > >| MClear
PyMOL Viewer
(all)
1FJF
S H L
Mouse: L M R
None Rota Move MovZ
Shft RotZ Move Clip
Ctrl +lb PkAt PkBd
CtSh lb Orig rb
Frame[1/ 1] 9/s
PyMOL>_
28,(chain A)
Start
PyMOL
PyMOL Viewer
PyMOL Molecular Graphic...
6:42 PM


Grab File Edit Capture Window Help
Thu 7:43 PM
PyMOL Viewer
(all)
ref
2fofc
m1
dist01
dist02
dist03
dist04
dist05
dist06
dist07
dist08
dist09
dist10
dist11
dist12
dist13
dist14
dist15
dist16
dist17
dist18
dist19
dist20
dist21
dist22
dist23
Mouse: L M R
None Rota Move MovZ
Shft RotZ Move Clip
Ctrl +lb PkAt PkBd
CtSh lb Orig rb
Frame[1/ 1] 4/s
PyMOL>_
Macintosh HD
Desktop (Mac OS 9)
pymol-0_59.tgz
PyMOL>11.614999771, 30.190002441, 20.704999924,\
PyMOL>27.582914352, 35.903408051, 1.000000000)
Scene: view updated.PyMOL>set nonbonded_size=0.6
Setting: nonbonded_size set to 0.60000.
ObjectMesh: updating "m1".
IsosurfVolume: Surface generated using 16155 lines.

tomato mitochondrial genome

Tools

- BLAST
- Clone Request

Methods

- Subcellular fractionation and mtDNA extraction
- Library construction and standard nomenclature
- In house pipeline for sequence analysis

eusol

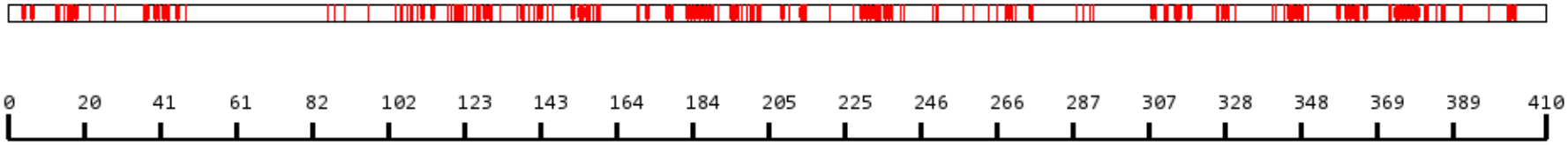
Lat-SOL

Backend

Background

The mitochondrion is the organelle within the eukaryotic cell that is primarily concerned with the synthesis of ATP in respiration. Current hypothesis traces it origins to an endosymbiotic event to form a cell lineage containing two independent genomes. Full comprehension of the mitochondrial (mt) genome structure is of fundamental importance not only for it evolutionary implications but also for a better understanding of biochemical processes that take place into this organelle in crop model species. Several mt genome sequences have been recently released from higher plants. We present here the tomato mitochondrial genome and a web-interface resource for plant mitochondrial comparative genomics.

Provisional Map

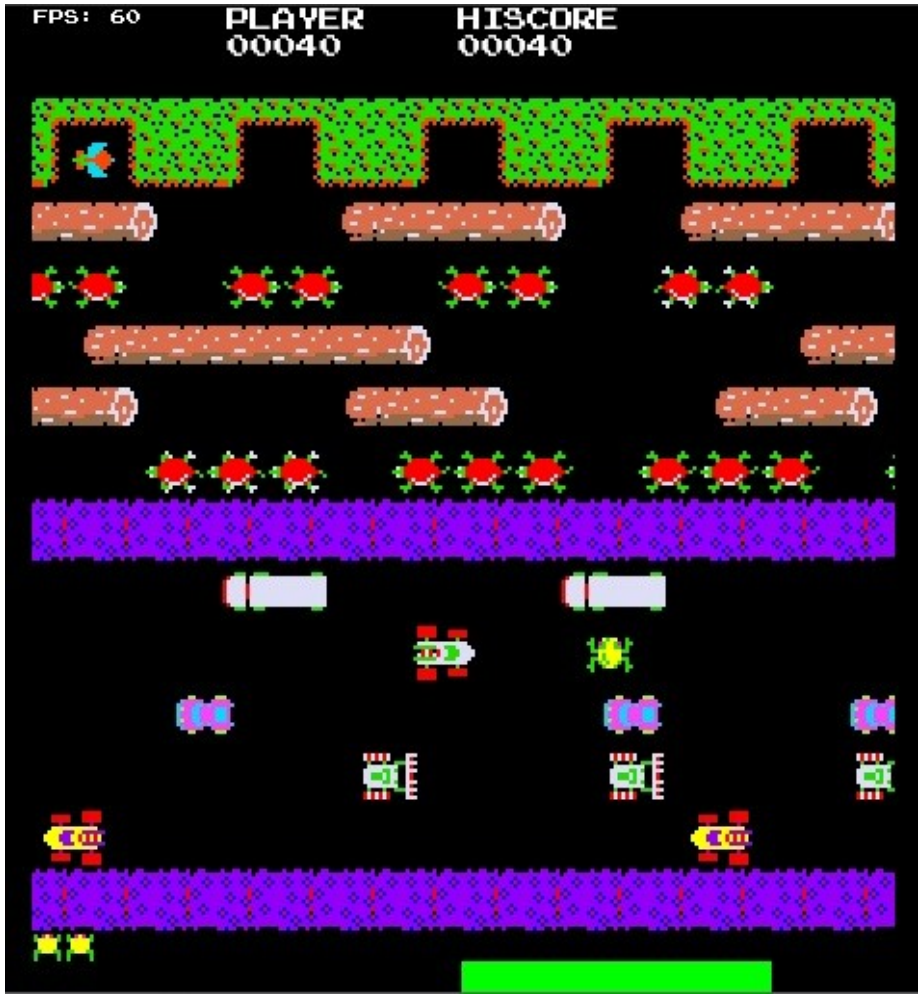


Click on the Graph to Zoom In

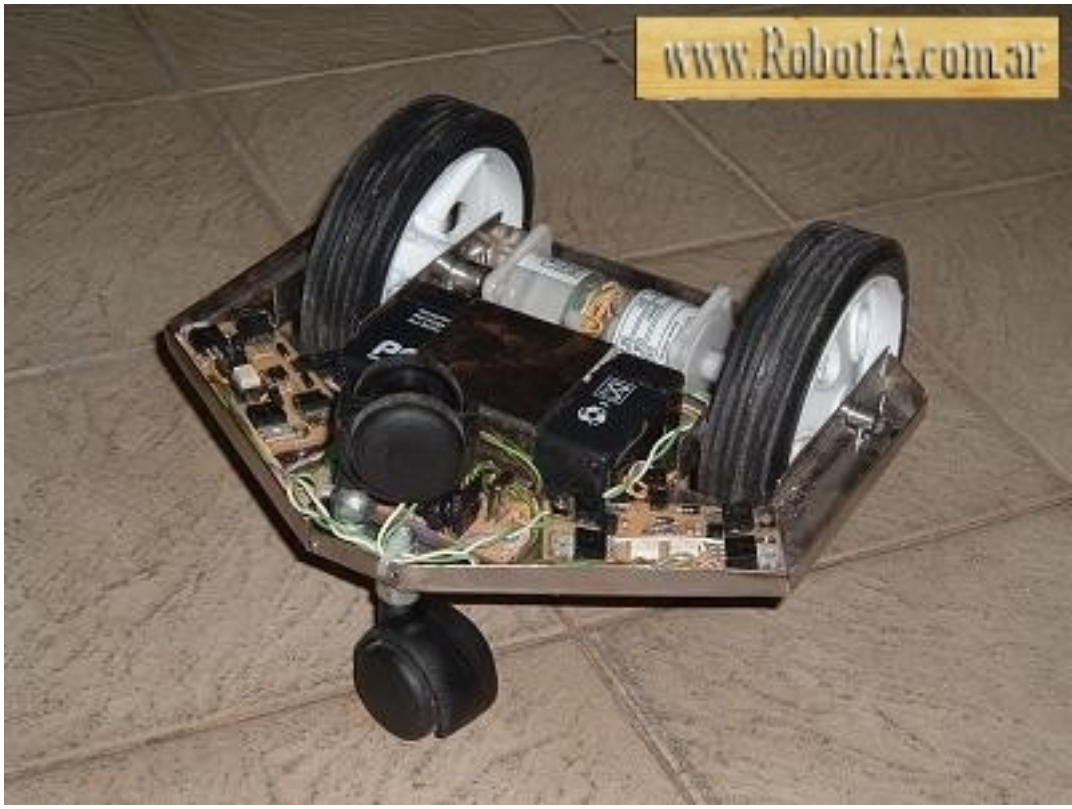
References

- Scotti N., Cardi T. Anda Marechal-Drouard L. (2001) Mitochondrial DNA and RNA isolation from small amounts of Potato tissue. Plant Molecular Biology Reporter 19;67a-67h.
- Gianniny C., Stoeva P., Cheely A. and Dimaculangan D. (2004) RAPD analysis of mtDNA from tomato flowers freee of nuclear DNA artifacts. Biotechniques 36:772-776.

Usos de Python: Juegos



Robots (made in Argentina)



www.robotia.com.ar

Tipo de datos: Primarios y derivados

Primarios (o primitivos): No necesitan de otro tipo de datos, como numericos (int, float, decimal, complex) y str (cadenas).

Derivados: Agrupan a alguno de los anteriores, como listas, diccionarios, tuplas, etc.
Se pueden subclasificar según distintos parámetros:

- Ordenados (o secuenciales) – Desordenados
- Mutables – Inmutables

```
>>> type(5)
<type 'int'>
>>> type(5.0)
<type 'float'>
>>> type(5 + 5.0)
<type 'float'>
>>> 5 + 5.0
10.0
>>> type(2+3j)
<type 'complex'>
>>> (2+3j).real
2.0
>>> (2+3j).imag
3.0
>>> type('Hola!')
<type 'str'>
>>> 'hola' + ' mundo!'
'hola mundo!'
>>> 'hela' + 2
Traceback (most recent call last):
  File "<pyshell#32>", line 1, in <module>
    'hela' + 2
TypeError: cannot concatenate 'str' and 'int' objects
>>> 'hela' + str(2)
'hela2'
```

str (String o Cadenas)

```
>>> 'Hola mundo!'
'Hola mundo!'
>>> a='Hola mundo!'
>>> len(a)
11
>>> a.lower()
'hola mundo!'
>>> a.count('o')
2
>>> a.find('H')
0
>>> a.find('mundo')
5
>>> a.find('e')
-1
>>> a.index(' ')
4
>>> a.index('e')
Traceback (most recent call last):
  File "<pyshell#52>", line 1, in <module>
    a.index('e')
ValueError: substring not found
>>> a.split(' ')
['Hola', 'mundo!']
```

http://docs.python.org/library/string.html

Datos ordenados: Listas

```
>>> mi_lista = [1,2,3]
>>> mi_lista.append(5)
>>> mi_lista
[1, 2, 3, 5]
>>> mi_lista.pop()
5
>>> mi_lista
[1, 2, 3]
>>> mi_lista + [4]
[1, 2, 3, 4]
>>> mi_lista
[1, 2, 3]
>>> mi_lista = mi_lista + [4]
>>> mi_lista
[1, 2, 3, 4]
>>> mi_lista.extend([5,6])
>>> mi_lista
[1, 2, 3, 4, 5, 6]
>>> mi_lista[0]
1
>>> mi_lista[3]
4
>>> mi_lista[3:5]
[4, 5]
>>> mi_lista[-2]
5
```

Mas sobre listas:

```
>>> variada = ['boga', 'cornalito', 'tararira']
>>> variada[2]
'tararira'
>>> variada[2][2:8]
'rarira'
>>> variada[2][2:]
'rarira'
>>> variada.append('pulpo')
>>> variada
['boga', 'cornalito', 'tararira', 'pulpo']
>>> variada.remove('cornalito')
>>> variada
['boga', 'tararira', 'pulpo']
>>> variada.sort()
>>> variada
['boga', 'pulpo', 'tararira']
>>> variada.index('pulpo')
1
>>> variada.index('pulpa')
Traceback (most recent call last):
  File "<pyshell#33>", line 1, in <module>
    variada.index('pulpa')
ValueError: list.index(x): x not in list
>>> 'pulpo' in variada
True
>>> 'pulpa' in variada
False
```

Diccionarios: Datos agrupados por clave-valor, sin orden.

```
>>> en2es = {'blue':'azul','red':'rojo','black':'negro'}
>>> en2es['blue']
'azul'
>>> en2es['azul']
Traceback (most recent call last):
  File "<pyshell#47>", line 1, in <module>
    en2es['azul']
KeyError: 'azul'
>>> 'blue' in en2es #verifico que una clave exista en 1 dict.
True
>>> en2es.keys()
['blue', 'black', 'red']
>>> en2es.values()
['azul', 'negro', 'rojo']
>>> en2es.items()
[('blue', 'azul'), ('black', 'negro'), ('red', 'rojo')]
>>> en2es.get('green','N/D')
'N/D'
>>> es2en = {}  #Diccionario vacio
>>> es2en['azul'] = 'blue' #Cargo un par clave-valor
>>> es2en
{'azul': 'blue'}
```

Ejemplo de diccionario

```
tom_map = { 1992: { 1:[ ('1A', 8.9), ('1B', 13.6), ('1C', 22.3), ('1D', 60.8), ('1E', 70.4), ('1F-
G', 93.6), ('1H', 111.7), ('1I', 129.2), ('1J', 10000)], 2:[ ('2A', 1.6), ('2B', 16.2), ('2C',
18.6), ('2D', 22.5), ('2E', 27.6), ('2F', 44.8), ('2G', 68), ('2H', 72.4), ('2I', 76.1), ('2J',
100.5), ('2K', 122.9), ('2L', 10000)], 3:[ ('3A', 24.2), ('3B', 30.4), ('3C', 54.8), ('3D', 61.1),
('3E', 64.4), ('3F', 97), ('3G', 98.4), ('3H', 108), ('3I', 100000)], 4:[ ('4A', 2), ('4B', 6.6),
('4C', 32.9), ('4D', 38), ('4E', 50), ('4F', 58.4), ('4G', 100.5), ('4H', 113.2), ('4I', 10000)],
5:[ ('5A', 4.6), ('5B', 17.2), ('5C', 42.8), ('5D', 44.6), ('5E', 72.7), ('5F', 75), ('5G', 84.9),
('5H', 92.3), ('5I', 10000)], 6:[ ('6A', 25), ('6B', 31.8), ('6C', 42), ('6D', 61.9), ('6E',
69.6), ('6F', 89.6), ('6G', 10000)], 7:[ ('7A', 3), ('7B', 11), ('7C', 21), ('7D', 36.8), ('7E',
52.6), ('7F', 70.6), ('7G', 75.7), ('7H', 10000)], 8:[ ('8A-B', 18.2), ('8C', 20.1), ('8D', 41.1),
('8E', 61.3), ('8F', 80.6), ('8G', 89.1), ('8H', 10000)], 9:[ ('9A', 8.9), ('9B', 22), ('9C',
28.9), ('9D', 39), ('9E', 56.4), ('9F', 57.4), ('9G', 64.2), ('9H', 69.1), ('9I', 79), ('9J',
102.6), ('9K', 10000)], 10:[ ('10A', 12), ('10B', 37.3), ('10C-D', 48.8), ('10E', 64.6), ('10F',
84.1), ('10G', 10000)], 11:[ ('11A', 20.8), ('11B', 32.3), ('11C', 45.4), ('11D', 59.3), ('11E',
79.9), ('11F', 83.3), ('11G', 83.8), ('11H', 10000)], 12:[ ('12A', 13.8), ('12B', 28.2), ('12C',
32.5), ('12D', 41), ('12E', 47.6), ('12F', 67.3), ('12G', 86), ('12H', 91.8), ('12I', 10000)]} ,
         2000 :{1:[ ('1A', 19.5), ('1B', 25), ('1C', 31.8), ('1D', 70), ('1E', 92.7), ('1F-G',
127.6), ('1H', 142), ('1I', 163), ('1J', 10000)], 2:[ ('2A', 4), ('2B', 13), ('2C', 16), ('2D-E',
31), ('2F', 45.1), ('2G', 81.2), ('2H', 85), ('2I', 90.1), ('2J', 116.1), ('2K', 143), ('2L',
10000)], 3:[ ('3A', 32), ('3B', 33), ('3C', 71.5), ('3D', 83), ('3E', 85), ('3F', 129), ('3G',
140), ('3H-I', 10000)], 4:[ ('4A-B', 12), ('4C', 46), ('4D', 56), ('4E', 72.5), ('4F', 75), ('4G',
101), ('4H', 124), ('4I', 10000)], 5:[ ('5A', 13.5), ('5B', 30), ('5C', 69), ('5D', 71.1), ('5E',
102), ('5F', 104), ('5G', 110.1), ('5H', 112), ('5I', 10000)], 6:[ ('6A', 33.5), ('6B', 38.6),
('6C', 50), ('6D', 71), ('6E', 81), ('6F', 96), ('6G', 10000)], 7:[ ('7A', 2), ('7B', 7), ('7C',
21.5), ('7D', 45.5), ('7E', 48), ('7F', 72.3), ('7G', 73), ('7H', 10000)], 8:[ ('8A', 2), ('8B',
23.8), ('8C', 30), ('8D', 40), ('8E', 57), ('8F', 68.3), ('8G', 84), ('8H', 10000)], 9:[ ('9A',
4), ('9B', 28), ('9C', 32), ('9D', 35), ('9E', 50.3), ('9F', 53.7), ('9G', 57.5), ('9H', 62),
('9I', 72.5), ('9J', 102), ('9K', 10000)], 10:[ ('10A', 11), ('10B', 43), ('10C-E', 61.5), ('10F',
80), ('10G', 10000)], 11:[ ('11A', 20.5), ('11B', 36.5), ('11C', 49), ('11D', 76), ('11E', 90),
('11F-G', 92), ('11H', 10000)], 12:[ ('12A', 21), ('12B', 32.5), ('12C', 38), ('12D', 55.3),
('12E', 68.5), ('12F-G', 114), ('12H', 117), ('12I', 10000)]}}
```

Conversión de datos: En Python siempre es explicito

```
>>> rgb=dict([('blue','#0000FF'),('black','#000000'),('red','#FF0000')])
>>> rgb['black']
'#000000'
>>> list(t1)
['sgn1545', 5, 45]
>>> lista = list('Hago 1 lista')
>>> lista
['H', 'a', 'g', 'o', ' ', '1', ' ', 'l', 'i', 's', 't', 'a']
>>> tuple(lista)
('H', 'a', 'g', 'o', ' ', '1', ' ', 'l', 'i', 's', 't', 'a')
>>> str(lista)
"['H', 'a', 'g', 'o', ' ', '1', ' ', 'l', 'i', 's', 't', 'a']"
>>> ''.join(lista)
'Hago una lista'
>>> 'Hago una lista'.split(' ')
['Hago', 'una', 'lista']
```

Estructuras de control de flujo

- `if: Condición`
- `for: Repetición`
- `while: Repetición`

if

```
if <expresion1>:
    <Instrucciones>
elif <expresion2>:
    <Instrucciones>
else:
    <Instrucciones>
```

```
if coord != 'N/A':
    year = int(coord[0][-4:])
```

for

```
for <var> in <iterable>:
    <instrucciones>
```

```
for x in [1, 3, 4]:
    print x
```

while

```
while <expresion>:
    <instrucciones>
```

```
while mi_set:
  print mi_set.pop()
```

Biblioteca estándar (Python standard library)

- Servicios del sistema, fecha y hora, subprocesos, sockets, i18n y l10n, base de datos, threads, formatos zip, bzip2, gzip, expresiones regulares, XML (DOM y SAX), Unicode, SGML, HTML, XHTML, email, manejo asincrónico de sockets, clientes HTTP, FTP, SMTP, NNTP, POP3, IMAP4, servidores HTTP, SMTP, debugger, random, curses, logging, compilador, decompilador, CSV, análisis lexicográfico, interfaz gráfica incorporada, matemática real y compleja, criptografía, introspección, unit testing, doc testing, etc., etc...

```
import os.path
import urllib2
import zipfile
import csv

url = 'http://www.genesdigitales.com/curso/table.zip'
fn = os.path.basename(url)
data = urllib2.urlopen(url)
datafile = open(fn,'w')
datafile.write(data.read())
datafile.close()
myfile = zipfile.ZipFile(fn)
unzip_fn = myfile.namelist()[0]
myfile.extractall()
listado = csv.reader(open(unzip_fn))
listado.next()
all_len = []
for line in listado:
    all_len.append(int(line[3])-int(line[2]))

print "Promedio: %s"%str(float(sum(all_len))/len(all_len))
```

Mas información sobre Python

www.python.org

www.diveintopython.org

http://openbookproject.net//thinkCSpy/

www.python.org.ar

http://python.org.ar/pyar/Tutorial

www.tinyurl.com/biopython