



**Introducción a la línea de comando y a la
programación para análisis bioinformáticos**



Florencia Grattarola
(flograttarola@gmail.com)





Guido van Rossum
(61 años)



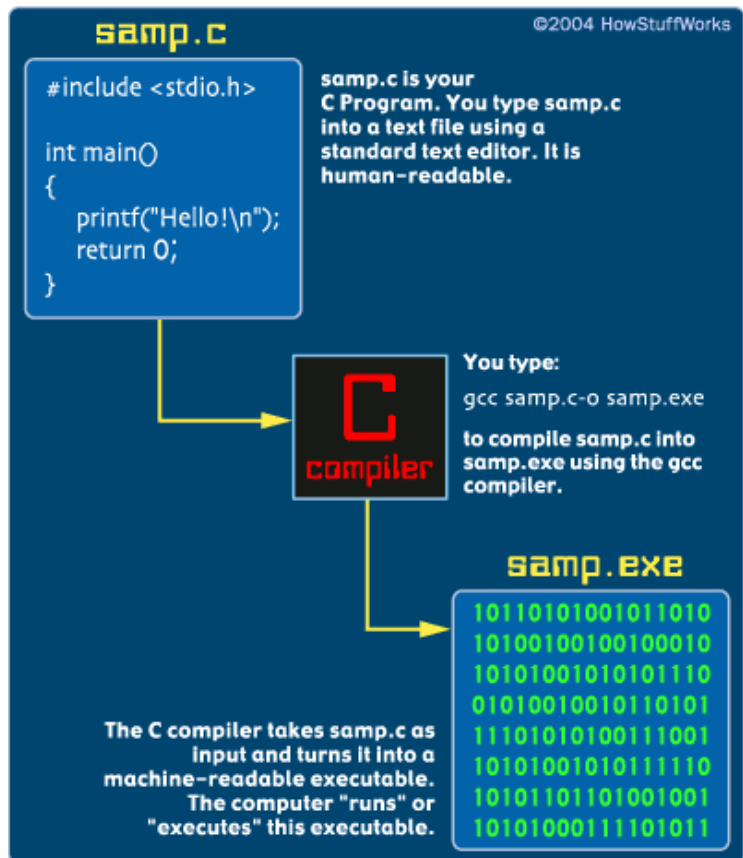
Monty Python



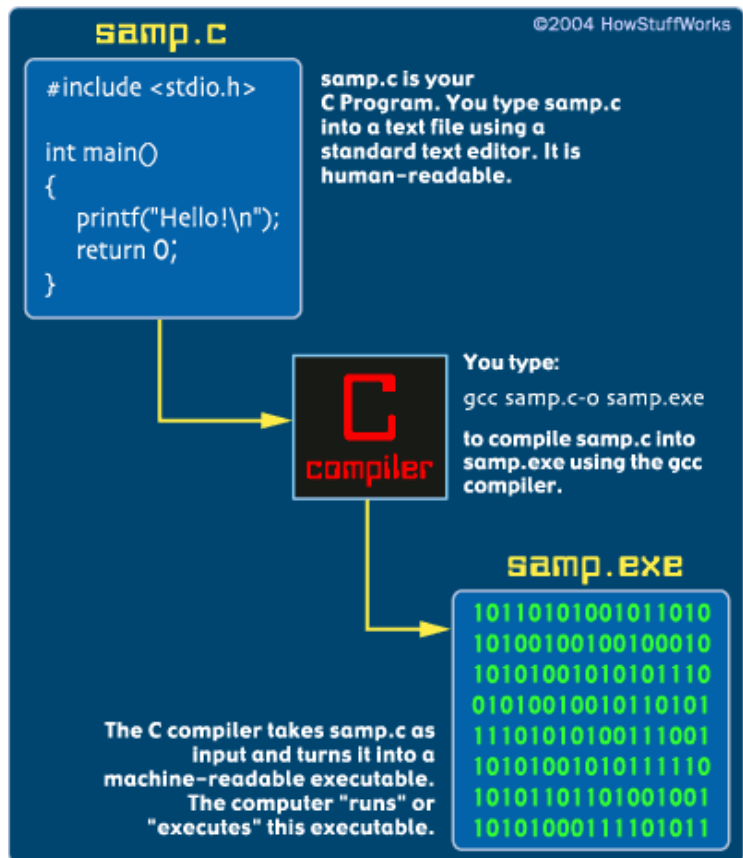
Código publicado 1991
Código abierto



5to lenguaje de
programación más popular
(2017)



Interpretedado



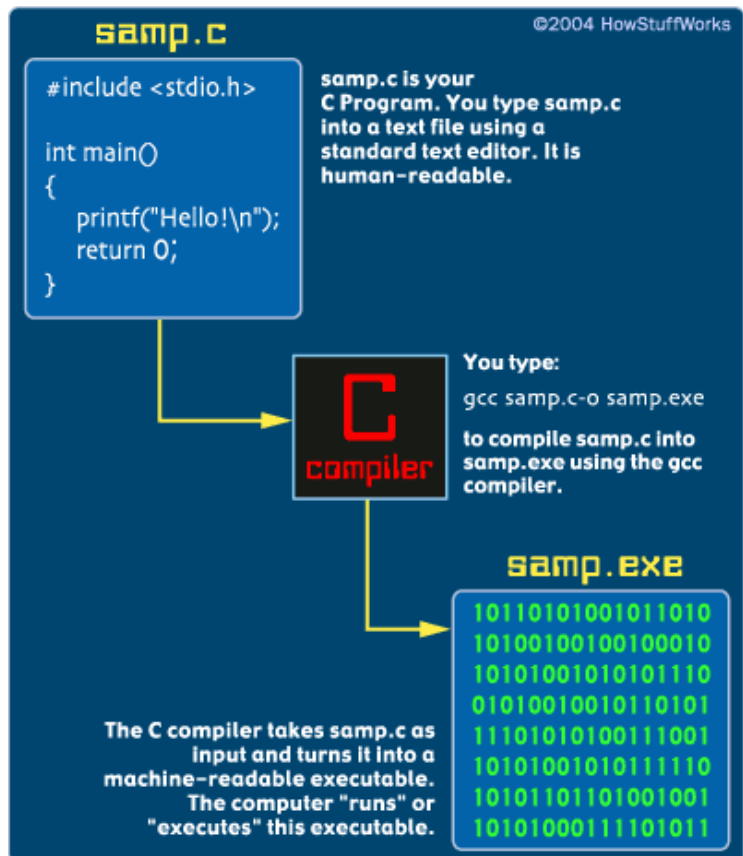
Curso Python : python – Konsole

Archivo Editar Ver Marcadores Preferencias Ayuda

```
florencia@FG:~/Documentos/Curso Python$ python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Curso Python : python

Interpretedado Interactivo

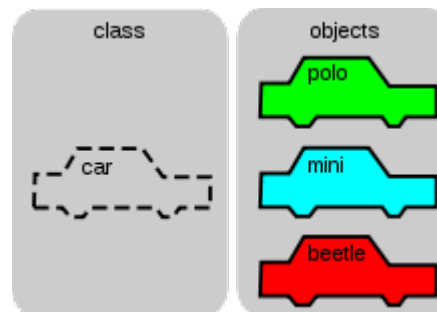


Curso Python : python - Konsole

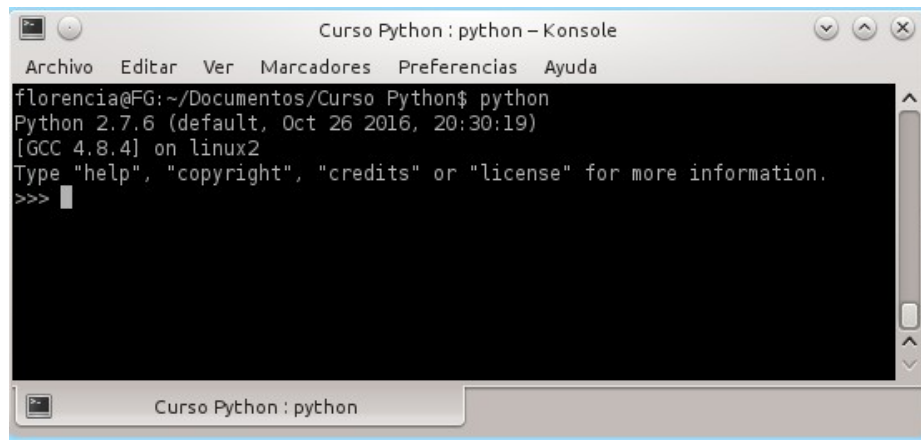
Archivo Editar Ver Marcadores Preferencias Ayuda

```
florencia@FG:~/Documentos/Curso Python$ python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

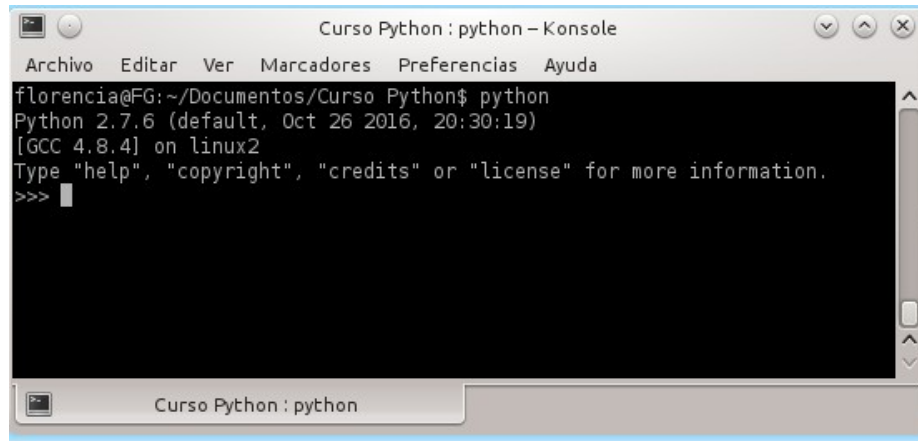
Curso Python : python



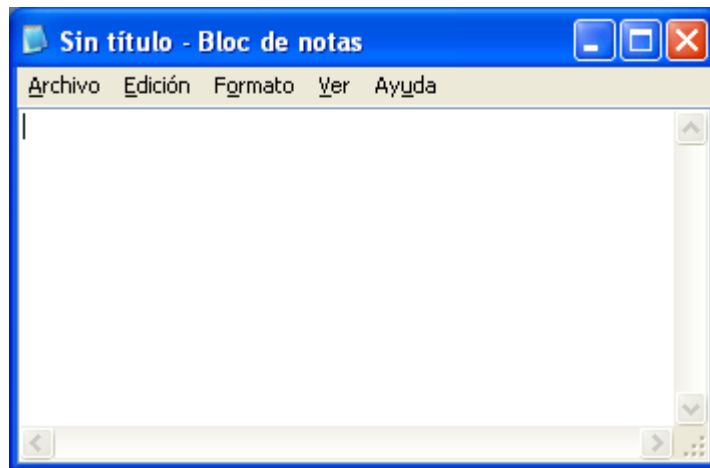
Interpretedado
Interactivo
Orientado a objetos



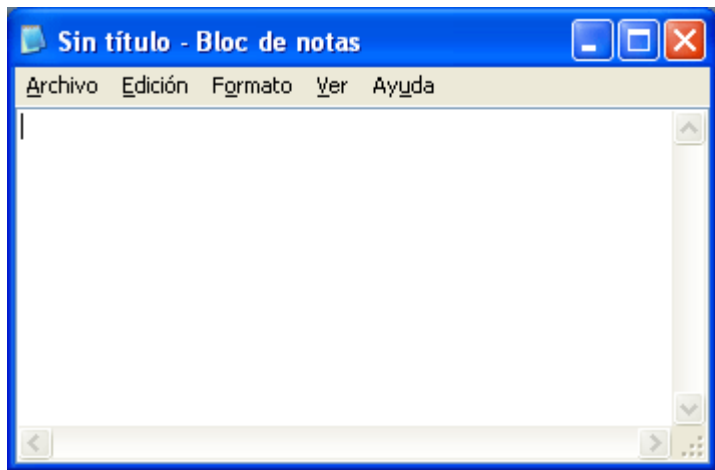
```
Curso Python : python - Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda
florecia@FG:~/Documentos/Curso Python$ python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```



```
Curso Python : python - Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda
florecia@FG:~/Documentos/Curso Python$ python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```



Editor de texto



Editor de texto



Kate

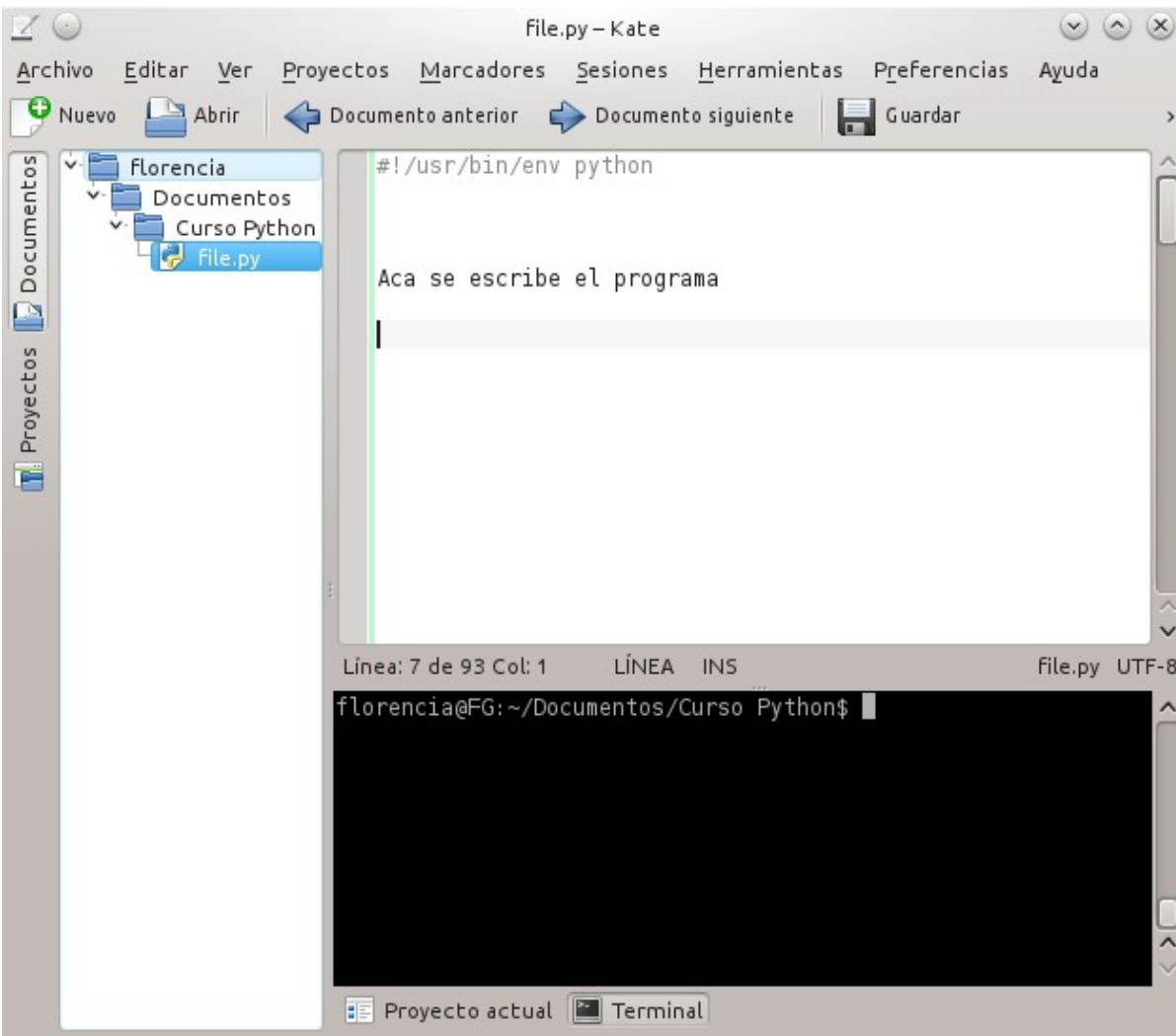


Notepad++

```

                                The
iLE88Dj. :jD8888Dj:
.LGtE888D.f8GjjjLE888E;
iE :8888Et. .G8888.
;I E888. ,8888.
D888. ,88888: 888
D888. 888 888888888 888 888 888
D888. 888 888 888 888 888 888
D888. 888 888 888 888888 888 888
D888. 88888 888P 888 88888 Y88b. ,d88P
888W. ,88888: "Y8888P88 888 Y888 "Y88888P"
W88W. ,8888:
W88W: ,88888: 88888b. 8888b. 88888b. ,d88b.
DG8D: ,88888: 888 "88b "88b 888 "88b d88b"88b
,88888: 888 888 ,d88888 888 888 888 888
,88888: 888 888 888 888 888 Y88 ,88P
,88888: 888 888 "Y88888 888 888 "Y88P"
E888i
tW88D                                Text Editor Homepage

```

file.py

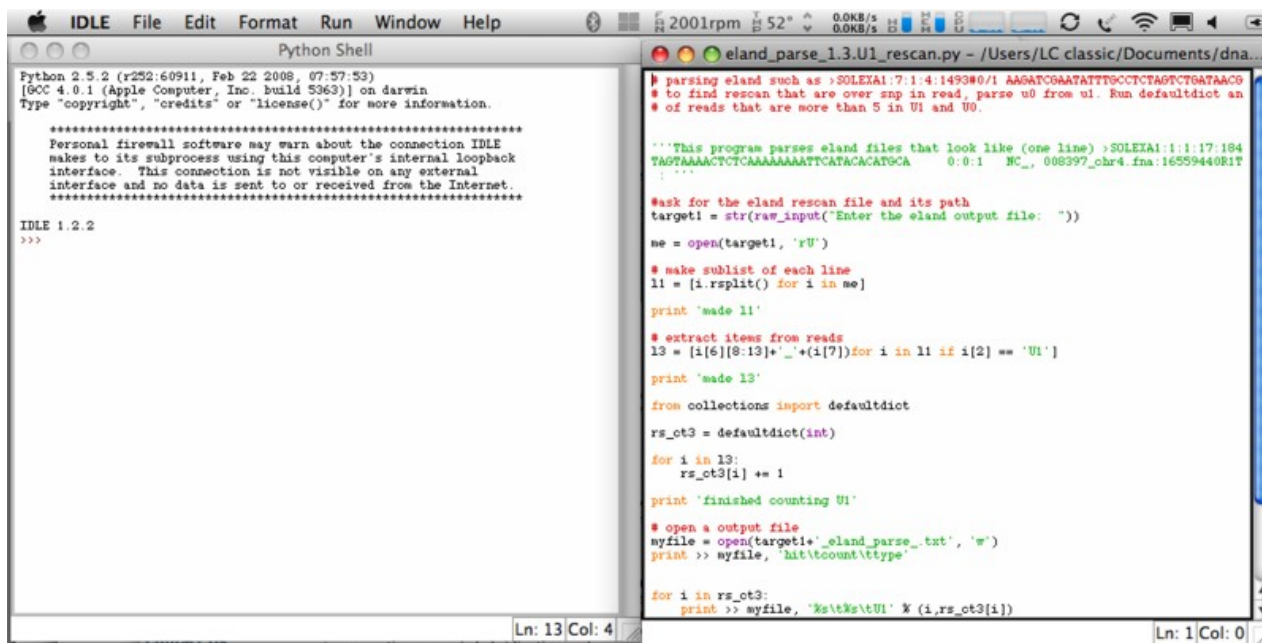
- Incluyendo:
`#!/usr/bin/python`
se vuelve ejecutable
- O se ejecuta con:
`> python file.py`

case and idention sensitive

IDLE

Integrated DeveLopment Environment

- Es una interfaz gráfica de usuario con dos tipos de ventana: Shell y Editor.
- Útil para usuarios Windows o Mac OS X



The screenshot shows the IDLE environment on a Mac OS X system. The top window is the 'Python Shell' with a menu bar (File, Edit, Format, Run, Window, Help) and a status bar at the bottom showing 'Ln: 13 Col: 4'. The shell displays the Python 2.5.2 version and a warning about the personal firewall. The bottom window is the 'eland_parse_1.3.U1_rescan.py' editor, showing a script that parses eland files. The editor has a menu bar and a status bar at the bottom showing 'Ln: 1 Col: 0'.

```
Python Shell
Python 2.5.2 (r252:60911, Feb 22 2008, 07:57:53)
[GCC 4.0.1 (Apple Computer, Inc. build 5363)] 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 1.2.2
>>>
```

```
eland_parse_1.3.U1_rescan.py - /Users/LC classic/Documents/dna...
# parsing eland such as : SOLEXIA1:7:1:4:1493#0/1 AAGATCGAATATTTCCTCTAAGTCTGATACG
# to find rescan that are over 500 in read, parse u0 from u1. Run defaultdict an
# of reads that are more than 5 in U1 and U0.

''' This program parses eland files that look like (one line) : SOLEXIA1:1:1:17:184
TAGTAAACTCTCAAAAAAATTCATACACATGCA 0:0:1 NC_, 008397_chr4.fna:16559440R1?
'''

#ask for the eland rescan file and its path
target1 = str(raw_input("Enter the eland output file: "))

me = open(target1, 'rU')

# make sublist of each line
l1 = [i.rstrip() for i in me]

print 'made l1'

# extract items from reads
l3 = [i[6][8:13]+'.'+i[17] for i in l1 if i[2] == 'U1']

print 'made l3'

from collections import defaultdict

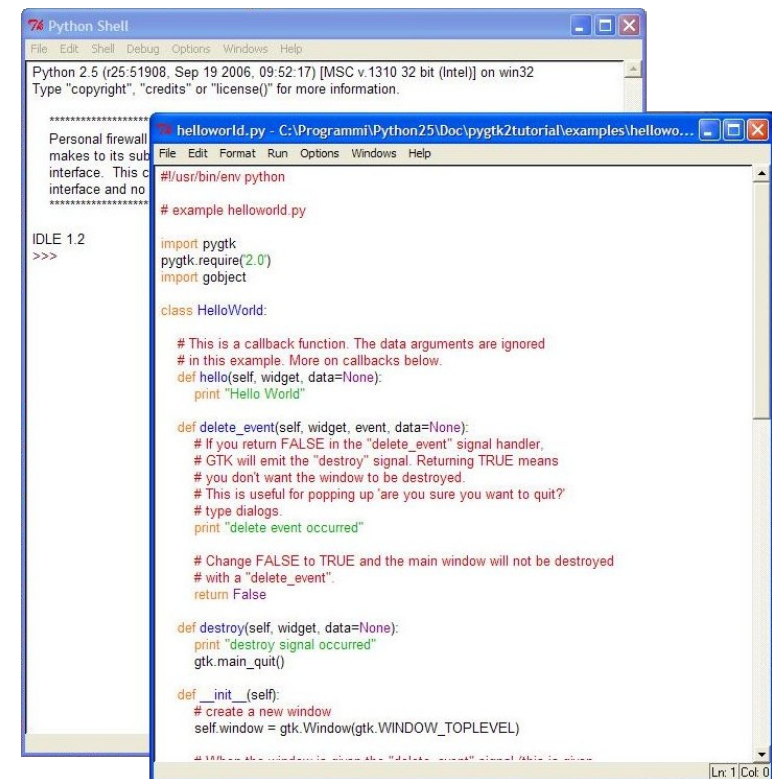
rs_ct3 = defaultdict(int)

for i in l3:
    rs_ct3[i] += 1

print 'finished counting U1'

# open a output file
myfile = open(target1+'.eland_parse.txt', 'w')
print >> myfile, 'hitcount\type'

for i in rs_ct3:
    print >> myfile, '%s\t%s\tU1' % (i,rs_ct3[i])
```



The screenshot shows the Python Shell and Editor windows. The top window is the 'Python Shell' with a menu bar (File, Edit, Shell, Debug, Options, Windows, Help) and a status bar at the bottom showing 'Ln: 1 Col: 0'. The shell displays the Python 2.5 version and a warning about the personal firewall. The bottom window is the 'helloworld.py' editor, showing a script that creates a 'HelloWorld' class using PyGTK. The editor has a menu bar and a status bar at the bottom showing 'Ln: 1 Col: 0'.

```
Python Shell
Python 2.5 (r25:51908, Sep 19 2006, 09:52:17) [MSC v.1310 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall
makes to its sub
interface. This c
interface and no
*****

helloworld.py - C:\Programmi\Python25\Doc\pygtk2tutorial\examples\hellowo...
File Edit Format Run Options Windows Help

#!/usr/bin/env python

# example helloworld.py

import pygtk
pygtk.require(2.0)
import gobject

class HelloWorld:

    # This is a callback function. The data arguments are ignored
    # in this example. More on callbacks below.
    def hello(self, widget, data=None):
        print "Hello World"

    def delete_event(self, widget, event, data=None):
        # If you return FALSE in the "delete_event" signal handler,
        # GTK will emit the "destroy" signal. Returning TRUE means
        # you don't want the window to be destroyed.
        # This is useful for popping up 'are you sure you want to quit?'
        # type dialogs.
        print "delete event occurred"

        # Change FALSE to TRUE and the main window will not be destroyed
        # with a "delete_event".
        return False

    def destroy(self, widget, data=None):
        print "destroy signal occurred"
        gtk.main_quit()

    def __init__(self):
        # create a new window
        self.window = gtk.Window(gtk.WINDOW_TOPLEVEL)

IDLE 1.2
>>>
```



Contenido

- **Tipo de datos**
- **Operadores**
- **Asignación (Statements and Expressions)**
- **Entrada y salida**

- Condicionales
- Instrucciones de repetición (Loops)

- File I/O
- Librerías y módulos



Tipo de datos

- Numéricos (enteros y reales)

```
>>> 14
14
>>> -1
-1
```

int

```
>>> 2.5
2.5
```

```
>>> 2e4
20000.0
```

```
>>> 2e-2
0.02
```

float



Tipo de datos

- Numéricos (enteros y reales)
- Lógicos (booleano)

```
>>> 14
14
>>> -1
-1
```

int

```
>>> 2.5
2.5
>>> 2e4
20000.0
>>> 2e-2
0.02
```

float

```
>>> True
True
>>> False
False
```

bool



Tipo de datos

- Cadena de caracteres (string)

```
>>> 'MNKMDLVADVAEKTDLSKAKATEVIDAVFA'
```

```
MNKMDLVADVAEKTDLSKAKATEVIDAVFA
```

```
>>> "AARHQGRGAPCGESFHHWALGADGGHGHQAQPPFRSSRLIGAERQPTSDCRQSLQQSPPC"
```

```
AARHQGRGAPCGESFHHWALGADGGHGHQAQPPFRSSRLIGAERQPTSDCRQSLQQSPPC
```

string



Tipo de datos

- Cadena de caracteres (string)
- Listas (built-in data type)

```
>>> 'MNKMDLVADVAEKTDLSKAKATEVIDAVFA'  
MNKMDLVADVAEKTDLSKAKATEVIDAVFA  
>>> "AARHQGRGAPCGESFHHWALGADGGHGHQAQPPFRSSRLIGAERQPTSDCRQSLQQSPPC"  
AARHQGRGAPCGESFHHWALGADGGHGHQAQPPFRSSRLIGAERQPTSDCRQSLQQSPPC
```

string

```
>>> [1, 2, 3, 4, 5]  
>>> [1, 2, 3]  
>>> [4, 5]
```

lst



Asignación

STATEMENT

Assignment

An *assignment statement* binds a name to an object. Assignment is denoted by a single equals sign:

name = value



Asignación

STATEMENT

Assignment

An *assignment statement* binds a name to an object. Assignment is denoted by a single equals sign:

name = value

( es diferente de )

(=): se usa para asignarle una valor a una variable
a = 0

(==): se usa como operador de comparación
si a == 0, entonces...



Asignación

```
>>> seq = 'AAAT'
```

```
>>> aaseq1 = 'MNKMDLVADVAEKTDLSKAKATEVIDAVFA'    # no value printed
```

```
>>> aaseq2 = 'AARHQGRGAPCGESFWHWALGADGGHGHQAQPPFRSSRLIGAERQPTSDCRQSLQ'
```

```
>>> aaseq3 = aaseq1
```

```
>>> a = b = c = 0
```

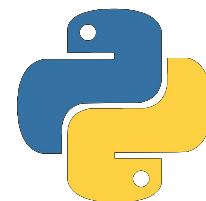
```
>>> a = a + 1      >>> a += 1
```

```
>>> list1 = [1,2,3]
```

```
>>> list2 = [4,5]
```

```
>>> list1 + list2
```

```
[1, 2, 3, 4, 5]
```



Operadores

- Numéricos

```
>>> 4 - 1
```

```
3
```

```
>>> 4 * 3
```

```
12
```

```
>>> 2 ** 10
```

```
1024
```

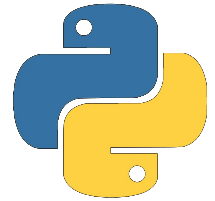
```
>>> 11 / 4
```

```
2.75
```

```
>>> 11 // 4
```

```
2
```

+, -, *, /,



Operadores

- Numéricos
- Lógicos

```
>>> 4 - 1
```

```
3
```

```
>>> 4 * 3
```

```
12
```

```
>>> 2 ** 10
```

```
1024
```

```
>>> 11 / 4
```

```
2.75
```

```
>>> 11 // 4
```

```
2
```

+, -, *, /,

```
>>> not True
```

```
False
```

```
>>> not False
```

```
True
```

```
>>> True and True
```

```
True
```

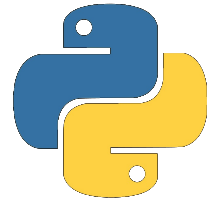
```
>>> True and False
```

```
False
```

```
>>> True or True
```

```
True
```

and, or, not



Operadores

- Numéricos
- Lógicos
- De comparación

```
>>> 4 - 1
```

```
3
```

```
>>> 4 * 3
```

```
12
```

```
>>> 2 ** 10
```

```
1024
```

```
>>> 11 / 4
```

```
2.75
```

```
>>> 11 // 4
```

```
2
```

`+, -, *, /,`

```
>>> not True
```

```
False
```

```
>>> not False
```

```
True
```

```
>>> True and True
```

```
True
```

```
>>> True and False
```

```
False
```

```
>>> True or True
```

```
True
```

`and, or, not`

`< menor`

`<= menor o igual`

`> mayor`

`>= mayor o igual`

`!= distinto`

`== igual`

`==, !=, <, >, <=, >=`



Operadores

- En strings

```
>>> 'TA' * 12  
'TATATATATATATATATATATATA'
```

```
>>> 6 * 'TA'  
'TATATATATATA'
```

```
>>> 'AC' + 'TG'  
'ACTG'
```

```
>>> 'aaa' + 'ccc' + 'ttt' + 'ggg'  
'aaaccctttggg'
```

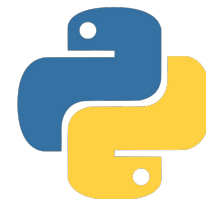
```
>>> 'TATA' in 'TATATATATATATATATATATATA'  
True
```

```
>>> 'AA' in 'TATATATATATATATATATATATA'  
False
```

```
>>> 'AA' not in 'TATATATATATATATATATATATA'  
True
```

and, or, not

+, -, *, /,



Operadores

- Slicing

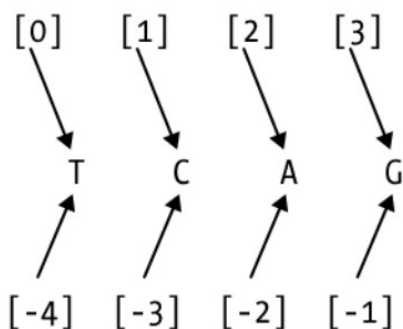


Figure 1-1. Index positions in strings

```
>>> 'MNKMDLVADVAEKTDL SKAKATEVIDAVFA'[0]
'M'
>>> 'MNKMDLVADVAEKTDL SKAKATEVIDAVFA'[1]
'N'
```

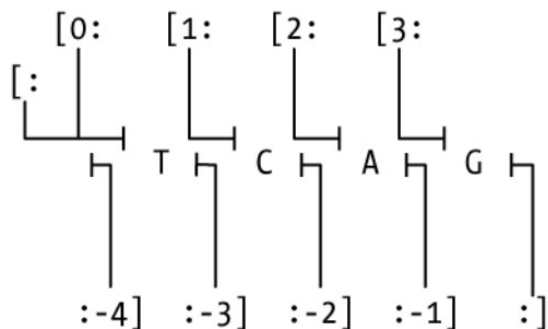


Figure 1-2. String slicing

```
>>> 'MNKMDLVADVAEKTDL SKAKATEVIDAVFA'[1:4]
'NKM'
```



Operadores

- Slicing

AGCTAGCATCGATCGATCTAGCTAGCT

Los dos primeros codones de la cadena?

El último codón?

El segundo y tercer codón?

Todos los codones menos el último?



Contenido

- ~~Tipo de datos~~
- ~~Operadores~~
- ~~Asignación (Statements and Expressions)~~
- **Entrada y salida**
- **Condicionales**
- **Instrucciones de repetición (Loops)**

- File I/O
- Librerías y módulos



Salida

- **print()** - utilizada para desplegar datos en la salida



Salida

- **print()** - utilizada para desplegar datos en la salida

```
print '\n'
print 'In the town where I was born lived a man who sailed the sea'
print 'And he told us of his life\nIn the land of submarines\n'
frase_1 = '\tWe all live in '
frase_2 = '\ta yellow submarine '
print frase_1 + frase_2
print frase_2 * 2
print '\n'
```

Línea: 43 de 54 Col: 37 LÍNEA INS

```
In the town where I was born lived a man who sailed the sea
And he told us of his life
In the land of submarines

    We all live in  a yellow submarine
    a yellow submarine      a yellow submarine
```

florencia@FG:~/Documentos/Curso Python\$

```
a = 1
b = a
print a
c = b + a
print c
```

Útil para
chequear
resultados de
procesos



Entrada

- **raw_input()** - Ingresa una línea desde la entrada estándar
- **input()** - Ingresa una línea desde la entrada y la evalúa como una expresión (número)



Entrada

- **raw_input()** - Ingresa una línea desde la entrada estándar
- **input()** - Ingresa una línea desde la entrada y la evalúa como una expresión (número)

Ejercicio: Ingreso de datos personales





Entrada

- **raw_input()** - Ingresa una línea desde la entrada estándar
- **input()** - Ingresa una línea desde la entrada y la evalúa como una expresión (número)

Ejercicio: Ingreso de datos personales

```
#Ejemplo ingreso de datos
```

```
print 'Ingrese los datos solicitados: '  
nombre = raw_input('Cual es tu nombre: ')  
edad = input('Cual es su edad: ')  
color = raw_input('Y su color preferido: ')  
  
print nombre, 'tiene', edad, 'años y su color preferido es el ', color
```

```
Línea: 33 de 45 Col: 5 LÍNEA INS
```

```
florencia@FG:~/Documentos/Curso Python$ ./file.py  
Ingrese los datos solicitados:  
Cual es tu nombre: Florencia  
Cual es su edad: 29  
Y su color preferido: verde  
Florencia tiene 29 años y su color preferido es el  verde  
florencia@FG:~/Documentos/Curso Python$
```



Contenido

- ~~Tipo de datos~~
- ~~Operadores~~
- ~~Asignación (Statements and Expressions)~~
- ~~Entrada y salida~~

- **Condicionales**
- **Instrucciones de repetición (Loops)**

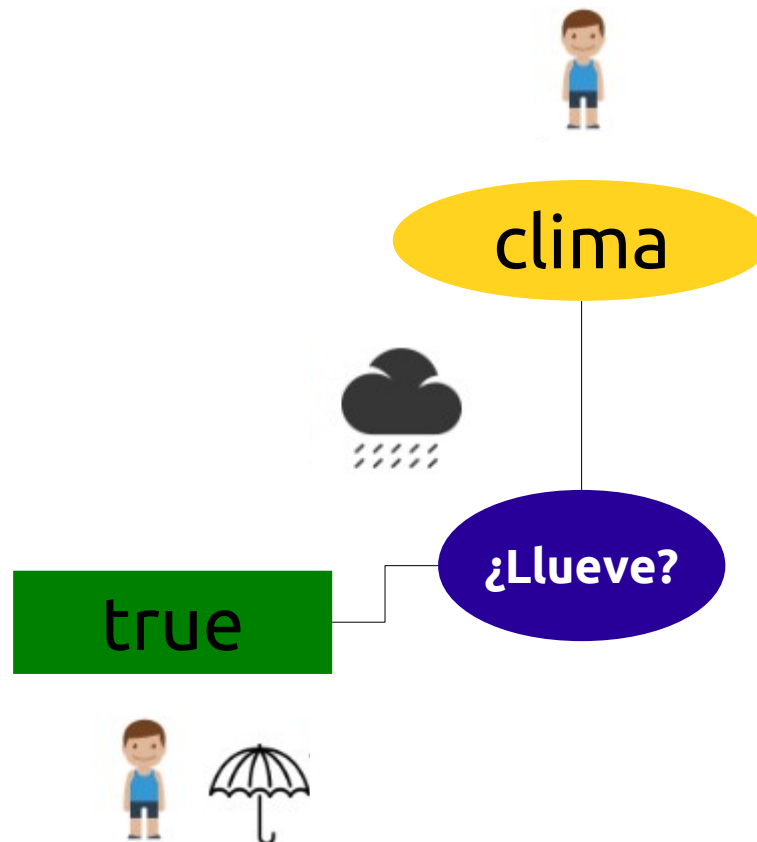
- File I/O
- Librerías y módulos



Condicionales

- **if**: si la expresión es verdadera, se ejecuta la declaración

```
if expression:  
    statements
```

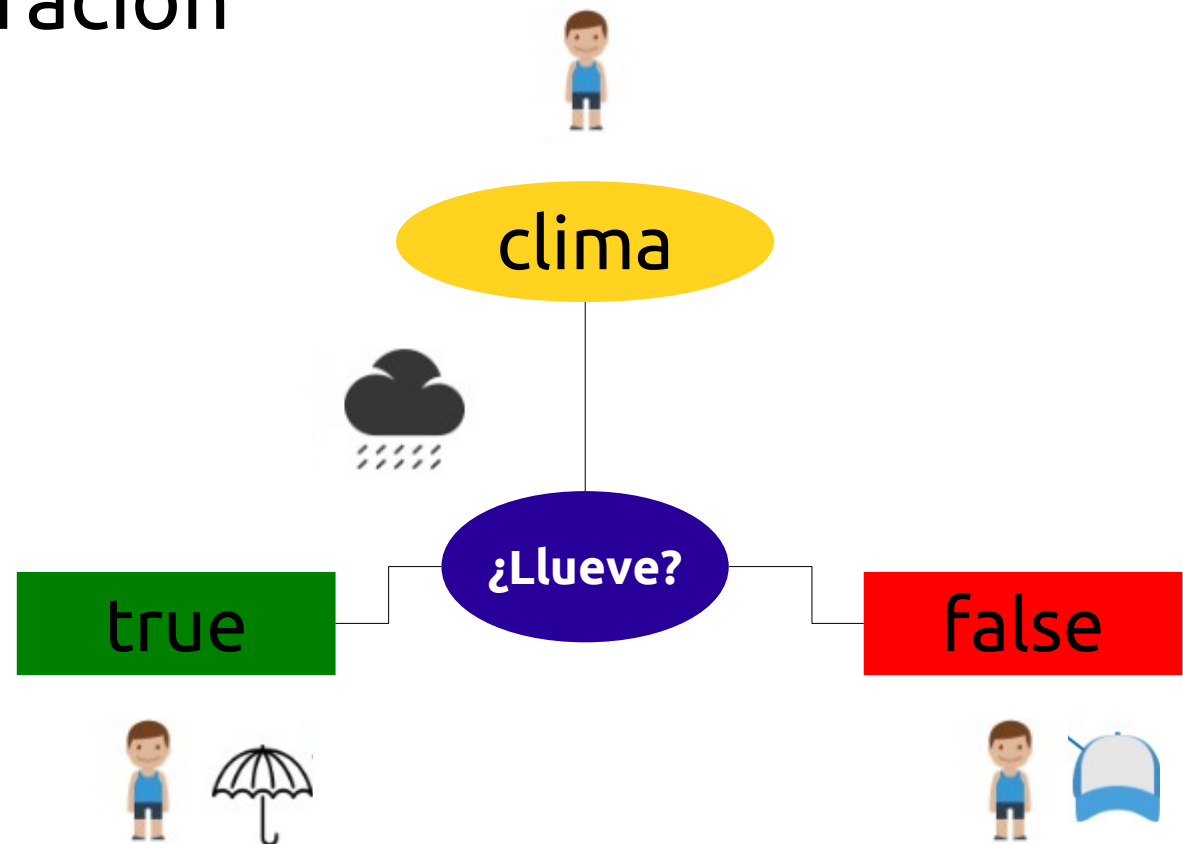




Condicionales

- **if/else:** si la expresión es verdadera se ejecuta lo declarado luego, si no, se ejecuta la segunda declaración

```
if expression:  
    statements1  
else:  
    statements2
```





Condicionales

- **if/else:** si la expresión es verdadera se ejecuta lo declarado luego, si no, se ejecuta la segunda declaración

Ejercicio: ¿cómo
determino si un numero
es par o impar?





Condicionales

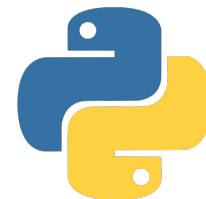
- **if/else:** si la expresión es verdadera se ejecuta lo declarado luego, si no, se ejecuta la segunda declaración

El
deter

```
numero = input('Numero: ')
if numero % 2 == 0:
    print 'El numero es par'
else:
    print 'El numero es impar'
```

Línea: 54 de 69 Col: 1 LÍNEA INS

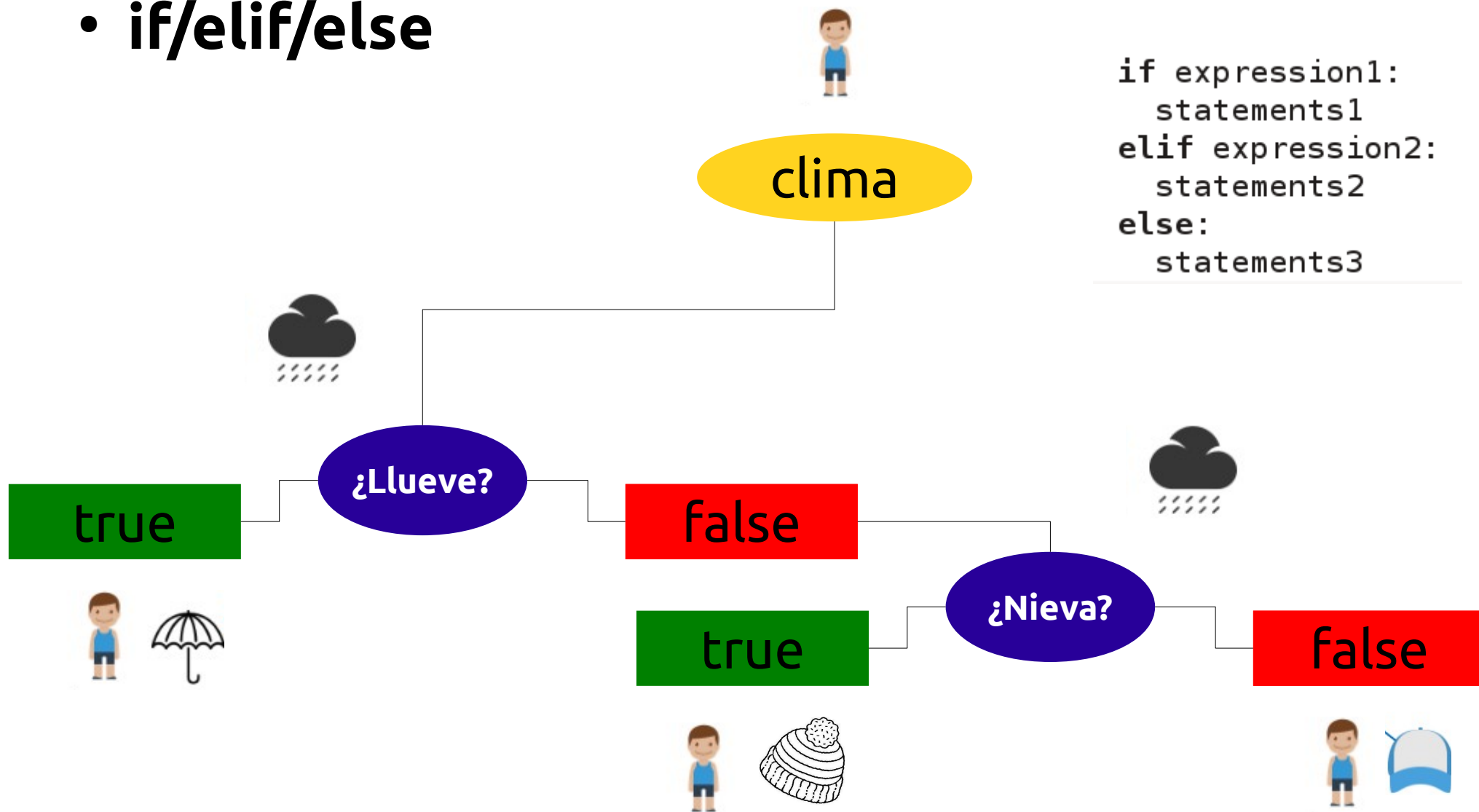
```
florencia@FG:~/Documentos/Curso Python$ ./file.py
Numero: 133
El numero es impar
florencia@FG:~/Documentos/Curso Python$ ./file.py
Numero: 234
El numero es par
florencia@FG:~/Documentos/Curso Python$ ./file.py
Numero: 123412312415668
El numero es par
florencia@FG:~/Documentos/Curso Python$
```

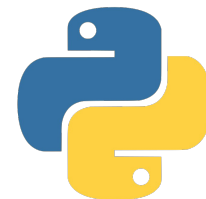


Condicionales

- if/elif/else

```
if expression1:  
    statements1  
elif expression2:  
    statements2  
else:  
    statements3
```





Condicionales

- **if/elif/else**

```
if expression1:  
    statements1  
elif expression2:  
    statements2  
else:  
    statements3
```

```
numero = input('Numero: ')\n▼ if numero == 0:\n    print 'El numero es cero'\n▼ elif numero % 2 == 0:\n    print 'El numero es par'\n▼ else:\n    print 'El numero es impar'\n\nLínea: 86 de 128 Col: 11    LÍNEA  INS\nflorescia@FG:~/Documentos/Curso Python$ ./file.py\nNumero: 2\nEl numero es par\nflorescia@FG:~/Documentos/Curso Python$ ./file.py\nNumero: 3\nEl numero es impar\nflorescia@FG:~/Documentos/Curso Python$ ./file.py\nNumero: 0\nEl numero es cero\nflorescia@FG:~/Documentos/Curso Python$ █
```

Instrucciones de repetición



while : repetición controlada por una condición

for: repetición asociada con una estructura de secuencia

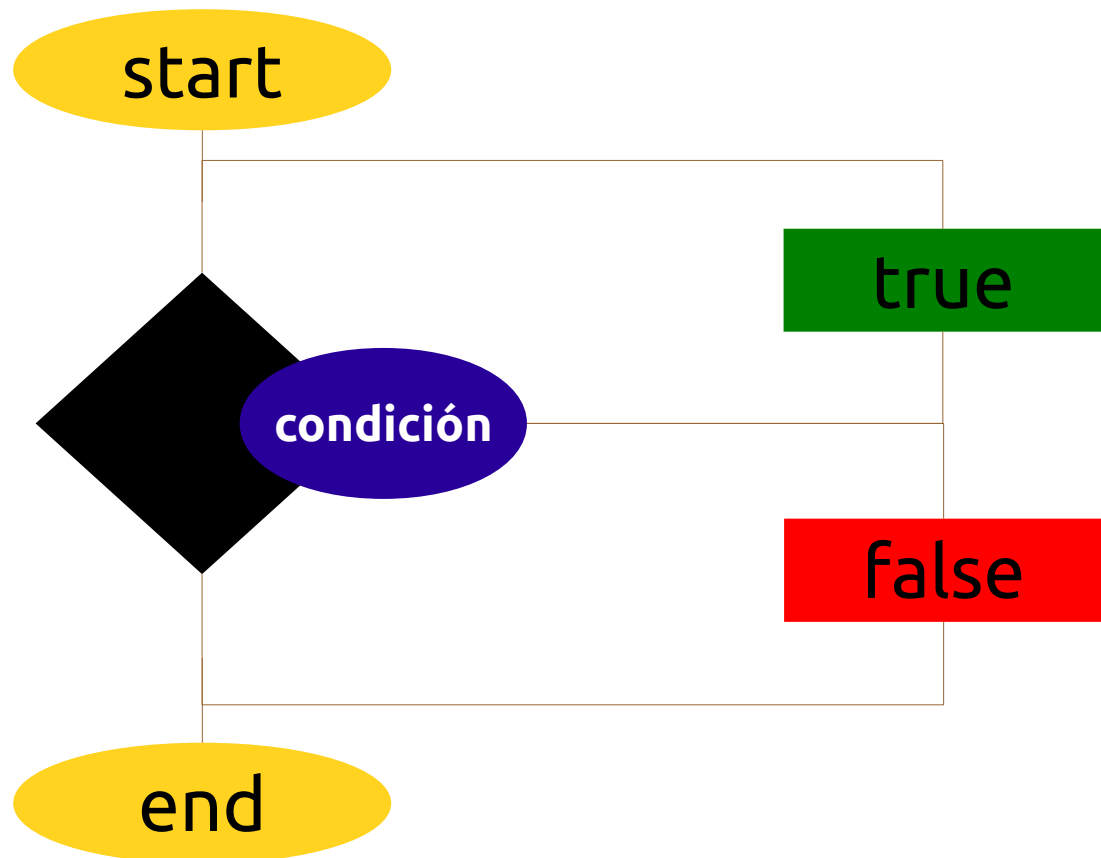


while

- Permite repetir la ejecución de un bloque mientras una condición se mantiene verdadera

`while expression:
statements`

while





while

- Permite repetir la ejecución de un bloque mientras una condición se mantiene verdadera

Ejercicio: Adivina un
numero del 1 al 10





while

- Permite repetir la ejecución de un bloque mientras una condición se mantiene verdadera

Ejercicio: Adivina un
numero del 1 al 10

```
num = 2
adivino = False
while not adivino:
    num_guess = input('Adivina, un numero del 1 al 10: ')
    if num_guess != num:
        print 'No adivinaste, intenta de nuevo'
    else:
        adivino = True
        print 'Adivinaste! el numero era: ', num
```

Línea: 34 de 45 Col: 1

LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
Adivina, un numero del 1 al 10: 3
No adivinaste, intenta de nuevo
Adivina, un numero del 1 al 10: 4
No adivinaste, intenta de nuevo
Adivina, un numero del 1 al 10: 2
Adivinaste! el numero era: 2
florencia@FG:~/Documentos/Curso Python$
```



for

- Sirve para recorrer todos los elementos de una secuencia y ejecutar un bloque asociado con cada elemento

```
for item in collection:  
    do something with item
```

for



for

- Sirve para recorrer todos los elementos de una secuencia y ejecutar un bloque asociado con cada elemento

```
for item in collection:  
    do something with item
```

for

Ejercicio: sumar todos los
numeros del 1 al 10





for

- Sirve para recorrer todos los elementos de una secuencia y ejecutar un bloque asociado con cada elemento

```
for item in collection:  
    do something with item
```

for

```
total = 0  
for number in range(1, 11):  
    total = total + number  
  
print total  
print (1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10)  
print range(1, 11)
```

Línea: 62 de 72 Col: 1 LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py  
55  
55  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
florencia@FG:~/Documentos/Curso Python$
```

os los
10



¿ while o for ?

- Puedo usar los dos, tengo que ver cuál me conviene.

while

for

Ejercicio: Cuenta regresiva
de 10 a 0





¿ while o for ?

- Puedo usar los dos, tengo que ver cuál me conviene.

while

for

```
numero = 10
▼ while numero >= 0:
    print numero
    numero = numero - 1
▼ for i in range(0, numero + 1):
    print numero - i
```

En cada vuelta

	num-i	impresión
i=1	10-1	9
i=2	10-2	8
i=3	10-3	7
i=4	10-4	6



Contenido

- ~~Tipo de datos~~
- ~~Operadores~~
- ~~Asignación (Statements and Expressions)~~
- ~~Entrada y salida~~

- ~~Condicionales~~
- ~~Instrucciones de repetición (Loops)~~

- **File I/O**
- **Librerías y módulos**



Files

- Los archivos son estructuras de datos que residen en el sistema.
- La manera de identificar un archivo es por su ruta en el sistema:

Unix /home/Documentos/file.txt

Windows c:\Documentos\file.txt



Abrir un archivo

- Es la operación que inicia el procesamiento de un archivo y lo deja preparado para operar con él

`file = open (ruta, modo)`

r	lectura
w	escritura/creación
a	escritura/extensión
r+	lectura y escritura



Lectura de un archivo

- Hay varias operaciones disponibles:

contenido: `file.read()` - *Lectura total*

línea: `file.readline()` - *Lectura de una línea*

líneas:

`file.readlines()` - *Lectura total como una lista de líneas*

`for linea in file` - *Iteración línea por línea*



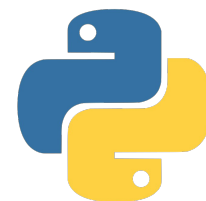
Lectura de un archivo

`file.read()`

`file.readline()`

`file.readlines()`

`for linea in file:`



Lectura de un archivo

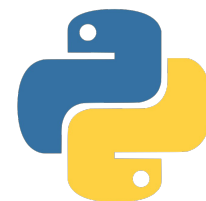
file.read()

```
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCCTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGGAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTAATAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAGAAGACGTTTTGCATTATGAG
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGACTAAACCAAACCCACCTTATAGCCAT
ACGACCAGGACTGTACTATGGCCAGTGCTCTGAAATCTGCGGATCTAACCACAGCTTTATACCCATTGTT
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTAATAGTATAA
```

```
file = open('coii_Clupus.fasta', 'r')
print file.read()
file.close()
```

Línea: 84 de 91 Col: 1 LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCCTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGGAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTAATAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAGAAGACGTTTTGCATTATGAG
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGACTAAACCAAACCCACCTTATAGCCAT
ACGACCAGGACTGTACTATGGCCAGTGCTCTGAAATCTGCGGATCTAACCACAGCTTTATACCCATTGTT
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTAATAGTATAA
```



Lectura de un archivo

file.readlines()

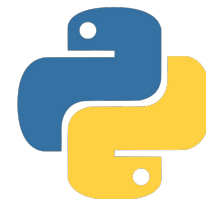
```
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGGAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTAATAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAGAAGACGTTTTGCATTGATGAG
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGACTAAACCAAACCAACCCCTTATAGCCAT
ACGACCAGGACTGTACTATGGCCAGTGCTCTGAAATCTGCGGATCTAACCACAGCTTTATACCCATTGTT
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTAATAGTATAA
```

```
file = open('coii_Clupus.fasta', 'r')
print file.readlines()
file.close()
```

Línea: 84 de 91 Col: 1 LÍNEA INS

File.py UTF-8

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
['>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome\n', 'ATGGCGTACCCATTTCAACTCGG
ATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC\n', 'ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATCA
TTTCACTAATATTGAC\n', 'TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGGAACAGTATGAACCATTCTACCCGCC\n', 'ATTATCCTA
ATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTAATAACCCCT\n', 'CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAG
CTATGAATATACTGACTATGAAGACTTAAA\n', 'CTTTGACTCCTACATAATCCCAACACAAGAATTAAGGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC\
n', 'CGAGTTGTCCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAGAAGACGTTTTGCATTGATGAG\n', 'CCGTTCCATCACTAGGTCTAAAACT
GACGCTATTCCAGGACGACTAAACCAAACCAACCCCTTATAGCCAT\n', 'ACGACCAGGACTGTACTATGGCCAGTGCTCTGAAATCTGCGGATCTAACCACAGCTT
TATACCCATTGTT\n', 'CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTAATAGTATAA\n', '\n']
```



Lectura de un archivo

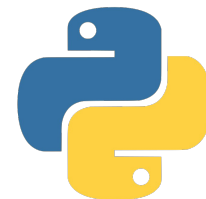
file.readline()

```
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCCTTAGTTCCTATATCATTCTACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGGAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTAAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAAGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAGAAGACGTTTTGCATTGATGAG
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGACTAAACCAAACCACCCCTTATAGCCAT
ACGACCAGGACTGTACTATGGCCAGTGCTCTGAAATCTGCGGATCTAACCACAGCTTTATACCCATTGTT
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTAATAGTATAA
```

```
file = open('coii_Clupus.fasta', 'r')
print file.readline()
file.close()
```

Línea: 88 de 91 Col: 20 LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
```



Lectura de un archivo

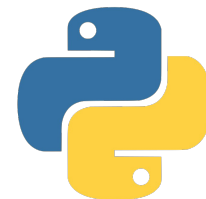
for linea in file:

```
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGAAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTATAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAAGACGTTTTCATTTCATGAG
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGACGAGTCTGAGTCTGAGTCTGAGTCTG
ACGACCAGGACTGACTATGGCCAGTGCTCTGAAATCTGCGGATCTGAGTCTGAGTCTGAGTCTGAGTCTG
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTA
```

```
file = open('coii_Clupus.fasta', 'r')
for linea in file:
    print linea
file.close()
```

Línea: 85 de 92 Col: 1 LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGAAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTATAACCCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGCCAGGAGAACTCCGACTATTAGAAGTAGACAAC
CGAGTTGTCCTCCCAATAGAAATAACCATCCGAATACTTATCTCTTCAAGACGTTTTCATTTCATGAG
```

Lectura de un archivo

for linea in file:

```
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGAAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTATAAACCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTGACTCCTACATAATCCCAACACAAGAATTAAGCCAGGAGA
CGAGTTGTCTCCCAATAGAAATAACCATCCGAATACTTATCTCT
CCGTTCCATCACTAGGTCTAAAACTGACGCTATTCCAGGACGAC
ACGACCAGGACTGACTATGGCCAGTGCTCTGAAATCTGCGGATC
CTTGAAATAGTCCCCCTATCTTACTTTGAGACCTGATCTGCCTTA
```

```
file = open('coii_Clupus.fasta', 'r')
for linea in file:
    print linea
file.close()

file = open('coii_Clupus.fasta', 'r')
for linea in file:
    print linea * 2
file.close()
```

Línea: 85 de 92

Línea: 76 de 93 Col: 1

LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
>gi|17737322:7034-7717 Canis lupus familiaris mitochondrion, complete genome
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGGCGTACCCATTTCAACTCGGATTACAGGACGCAACCTCCCCTATTATAGAGGAGCTACTTCATTTTC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
ATGACCATACACTAATAATTGTATTCTTAATCAGTTCTTTAGTTCTCTATATCATTTCACTAATATTGAC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGAAACAGTATGAACCATTCTACCCGCC
TACAAAATTAACCCATACAAGCACAATAGACGCACAAGAAGTGAAACAGTATGAACCATTCTACCCGCC
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTATAAACCCT
ATTATCCTAATCCTAATCGCTCTACCTTCCCTCCGAATCCTTTATATAATGGACGAAATTATAAACCCT
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
CTTTAACCGTGAAAACAATAGGCCACCAATGATACTGAAGCTATGAATATACTGACTATGAAGACTTAAA
```



Escritura de archivos

- Para escribir un archivo necesito abrirlo, escribir y luego al finalizar cerrarlo:

```
f = open('file.txt', 'w')
```

```
f.write('ACTGATAGCTAGACTATGCA')
```

```
f.close()
```



Escritura de archivos

- Para escribir un archivo necesito abrirlo, escribir y luego al finalizar cerrarlo:

```
f = open('file.txt', 'w')
```

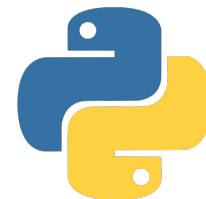
```
f.write('ACTGATAGCTAGACTATGCA')
```

```
f.close()
```

```
f = open('file.txt', 'w')  
f.write('ACTGATAGCTAGACTATGCA')  
f.close()
```

```
ACTGATAGCTAGACTATGCA
```

Línea: 1 de 1 Col: 1 Caracteres: 20 LÍNEA INS

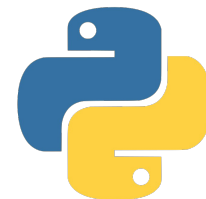


Lectura un archivo

Utilidades

```
LOCUS      NODE_119_length_284_cov_26.3684_ID_237          284 bp    DNA      linear
DEFINITION Contig NODE_119_length_284_cov_26.3684_ID_237 from Salmonella
            enterica UY054
ACCESSION   unknown
FEATURES             Location/Qualifiers
     source             1..284
         >              /mol_type="genomic DNA"
         >              /db_xref="taxon: 90371"
         >              /genome_md5=""
         >              /project="flograt_90371"
         >              /genome_id="90371.127"
         >              /organism="Salmonella enterica UY054"
BASE COUNT      76 a      94 c      66 g      48 t
ORIGIN
      1 accgcgaaca tcaaaggctc gactcaggct tcccgtaacg ctaacgacgg tatctccatt
     61 gcgcagacca ctgaaggcgc gctgaacgaa atcaacaaca acctgcagcg tgtgcgtgaa
    121 ctggcggttc agtctgctaa cagcaccaac tcccagtcgt acctcgactc catccaggct
    181 gaaatcacc agcgccctgaa cgaaatcgac cgtgtatccg gccagactca gtccaacggc
    241 gtgaaagtcc tggcgcagga caacaccctg accatccagg ctca
```

- Combinar archivos
- Extraer datos
- Realizar cálculos



Lectura un archivo

>UY099

```
GTGGCTGAGGATAAACATCTGCATCACTGGCTGGATATGGCGCGCGAGCGCATTCAATTTTCAGGGGTTACCGGCGCGTATCTGCTGGGTAGG
CCTGGAGTGGCGGCAAAAACTGGGGCTGGCGTTCAACGAAATGGTGCCTTGC GGCGAGGTATCCGCGCCCATTGTGATTGGCCGCGATCACC
TGGATTCCGGCTCTGTGCGCCAGCCCTAACCGTGAAACCGAAGCGATGCGCGACGGTCCGACGCGGTTCCGACTGGCCGCTGTTAAATGCG
TTGCTGAATACCGCCAGCGGGGCGACATGGGTATCGCTCCATCATGGCGGCGGGGTGGGAATGGGGTTTTCGCAACACGCCGCTATGGTGAT
TGTCTGTGATGGCACTGACGAGGCCGCCGCGCTATTCGCCGCGTGTACACAACGATCCGGCGACGGGCGTCATGCGCCATGCCGATGCCG
GATATGATCTCGCGGTGGAATGCGCTGTTGAGCAAGGTCTGAATTTACCGATGGTTGCGGCGACGCAGGGGAAAGGCTGA
```

>UY074

```
GTGGCTGAGGATAAACATCTGCATCACTGGCTGGATATGGCGCGCGAGCGCATTCAATTTTCAGGGGTTACCGGCGCGTATCTGCT
CCTGGAGTGGCGGCAAAAACTGGGGCTGGCGTTCAACGAAATGGTGCCTTGC GGCGAGGTATCCGCGCCCATTGTGATTGGCCGCG
TGGATTCCGGCTCTGTGCGCCAGCCCTAACCGTGAAACCGAAGCGATGCGCGACGGTCCGACGCGGTTCCGACTGGCCGCTGTT
TTGCTGAATACCGCCAGCGGGGCGACATGGGTATCGCTCCATCATGGCGGCGGGGTGGGAATGGGGTTTTCGCAACACGCCGCTA
TGTCTGTGATGGCACTGACGAGGCCGCCGCGCTATTCGCCGCGTGTACACAACGATCCGGCGACGGGCGTCATGCGCCATGCC
GATATGATCTCGCGGTGGAATGCGCTGTTGAGCAAGGTCTGAATTTACCGATGGTTGCGGCGACGCAGGGGAAAGGCTGA
```

	UY051	UY052	UY053	UY054	UY055	UY056	UY057	UY059
1 > 1	1	1	1	1	1	1	1	1
10 > 1	1	1	1	1	1	1	1	1
100 > 0	0	0	0	0	0	0	0	0
1000 > 1	1	1	1	1	1	1	1	1
1001 > 1	1	1	1	1	1	1	1	1
1002 > 1	1	1	1	1	1	1	1	1
1003 > 1	1	1	1	1	1	1	1	1
1004 > 0	0	0	0	0	0	0	0	0
1005 > 0	0	0	0	0	0	0	0	0
1006 > 0	1	0	0	0	0	1	0	0
1007 > 1	1	1	1	1	1	1	1	1
1008 > 0	0	0	0	0	0	0	0	0
1009 > 1	1	1	1	1	0	0	0	0
101 > 1	1	1	1	1	1	1	1	1
1010 > 1	1	1	1	1	1	1	1	1
1011 > 1	1	1	1	1	1	1	1	1
1012 > 1	1	1	1	1	1	1	1	1
1013 > 1	1	1	1	1	1	1	1	1
1014 > 1	1	1	1	1	1	1	1	1
1015 > 1	1	1	1	1	1	1	1	1



Librerías

- Una librería o módulo es un componente que provee definiciones de funciones, variables o clases relativas a un tema común.
- Permiten (re)utilizar facilidades ya definidas por otros sin necesidad de programarlas
- Se pueden escribir módulos propios de manera sencilla



Librerías

- Conocer el largo de caracteres de una cadena
len

```
cadena = 'ACGCAGCATAGCTACTATCAGCTAATCGACCGCTAGCAG'
largo = 0
for letra in cadena:
    largo = largo + 1
print largo
```

```
cadena = 'ACGCAGCATAGCTACTATCAGCTAATCGACCGCTAGCAG'
largo = len(cadena)
print largo
```



Librerías

- Conocer la cantidad de veces que aparece un caracter en una cadena *count*

```
cadena = 'ACGCAGCATAGCTACTATCAGCTAATCGACCGCTAGCAG'
cantidad_A = 0
for letra in cadena:
    if letra == 'A':
        cantidad_A = cantidad_A + 1
print cantidad_A
```

```
cadena = 'ACGCAGCATAGCTACTATCAGCTAATCGACCGCTAGCAG'
cantidad_A = cadena.count('A')
print cantidad_A
```




Librerías

- Para utilizar un módulo predefinido se debe importar en nuestro programa:

```
import nombre-módulo
```

- Por ejemplo:
 - import math



Librerías

- Raíz de un número

```
x= 2
raiz = x**0.5

print raiz
```

Línea: 126 de 142 Col: 1 LÍNEA INS

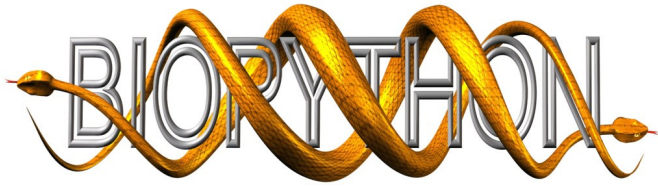
florencia@FG:~/Documentos/Curso Python\$./file.py
1.41421356237

```
import math

x= 2
raiz = math.sqrt(x)
print raiz
```

Línea: 130 de 142 Col: 1 LÍNEA INS

florencia@FG:~/Documentos/Curso Python\$./file.py
1.41421356237

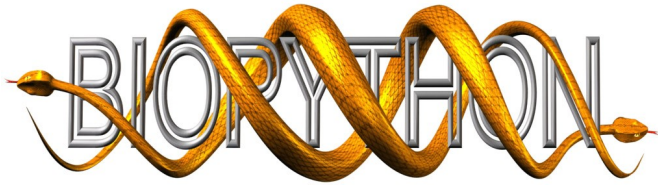


BioPython



Conjunto de herramientas para bioinformática y biología molecular:

- Clase '*sequence*' para guardar secuencias, IDs y distintas características (*features*)
- Interfaces con programas como clustalw, blast, primer3, y más.
- Herramientas para realizar operaciones comunes con secuencias: traducción, transcripción, Tm, peso molecular.



BioPython



- Ejemplo

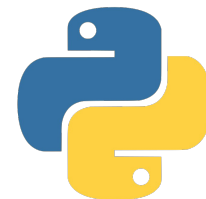
```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC

secuencia_dna = Seq('AAGATGCATCGATGCTAGATCGTA', IUPAC.unambiguous_dna)
rna = secuencia_dna.transcribe()
secuencia_aa = rna.translate()

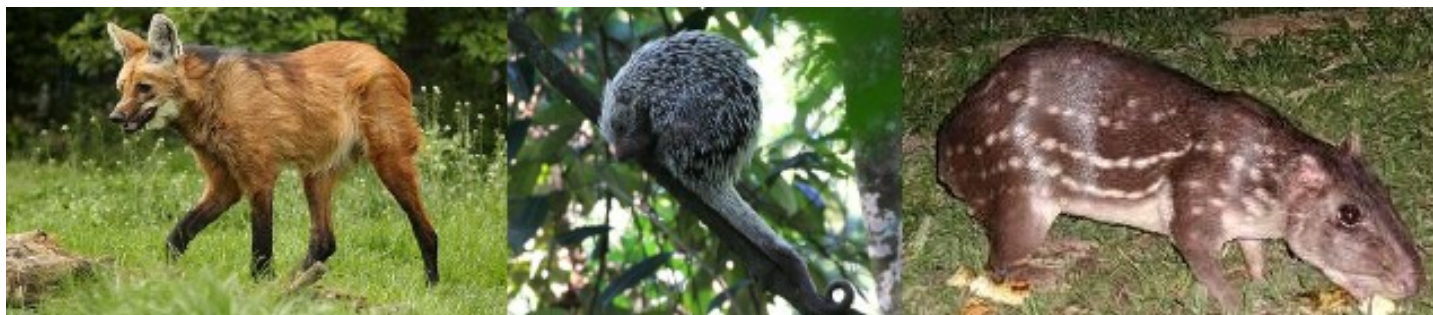
print secuencia_dna
print rna
print secuencia_aa
```

Línea: 105 de 117 Col: 1 LÍNEA INS

```
florencia@FG:~/Documentos/Curso Python$ ./file.py
AAGATGCATCGATGCTAGATCGTA
AAGAUGCAUCGAUGCUAGAU CGUA
KMHRC*IV
```

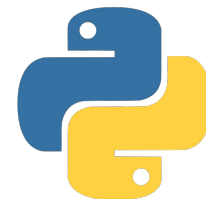


Ejemplo

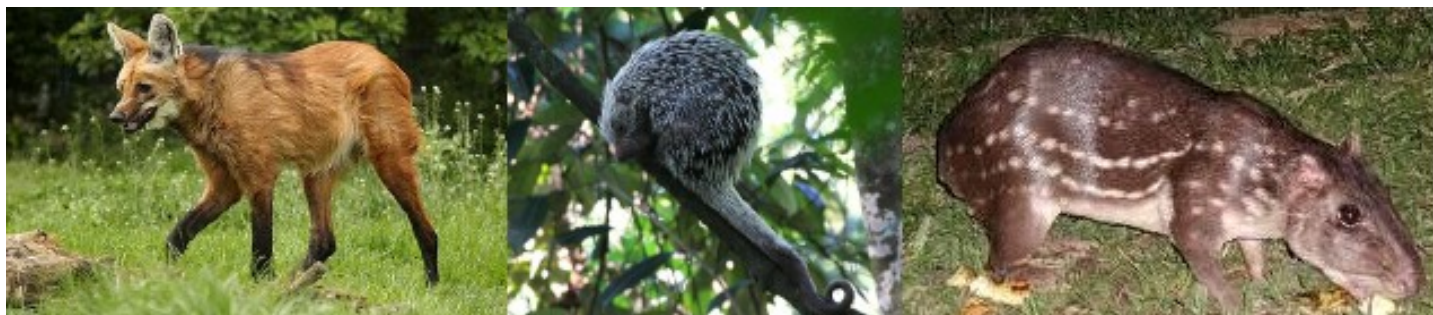


Conocer el ensamble de mamíferos de Paso Centurión a través de muestras colectadas de manera no invasiva.

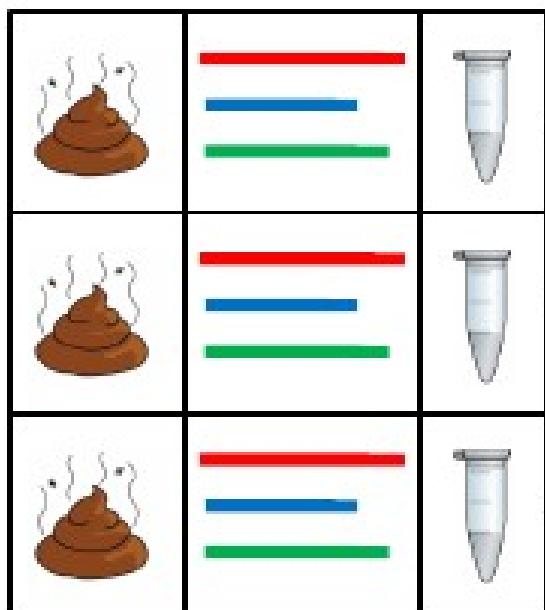




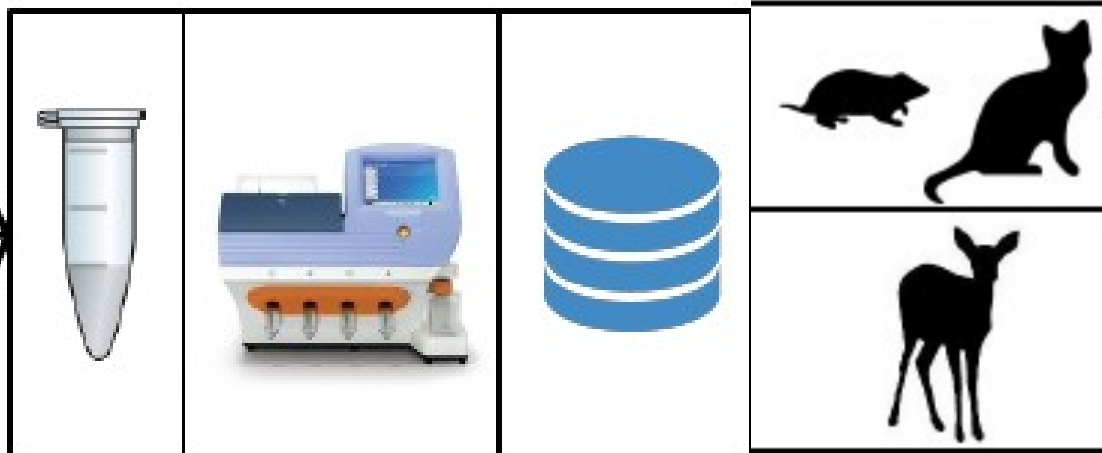
Ejemplo



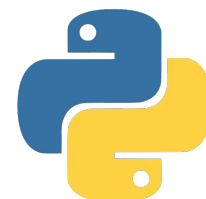
unión de productos



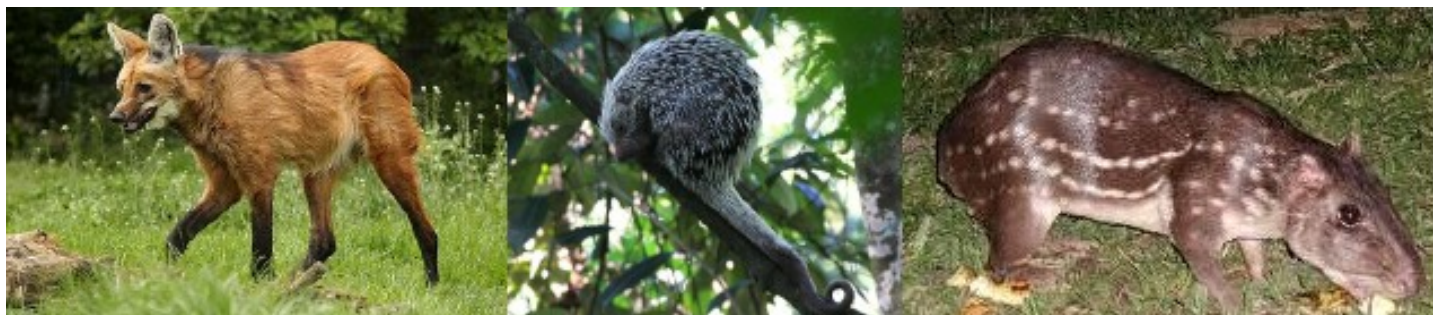
secuenciación
Ion Torrent™



identificación
taxonómica



Ejemplo



lecturas de
secuenciación

limpieza

```
ACTATAGCTATCGATC
TAGCTATCGATAAATT
CGCTATCGAT
```

SICKLE

Q>20
largo > 50pb

alineamiento

reporte



secuencias de referencia

Taxonomic Groups [List]

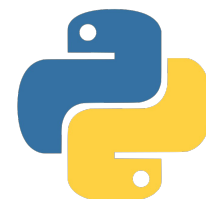
- mammals (30862)
 - placentals (29474)
 - even-toed ungulates (14898)
 - whales & dolphins (13831)
 - Balaenopteridae (12697)
 - more... (1134)
 - more... (1067)
 - rodents (2552)
 - Muridae (1620)
 - more... (1922)
 - bats (3222)
 - Vespertilionidae (1138)
 - more... (12061)



especies

nº de lecturas
por especies





Ejemplo



Ejemplo de archivo FASTQ:

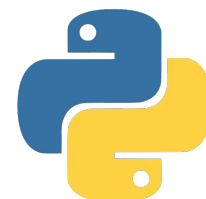
```
@OHYN0:00037:00271
CACCCAAAGCTGATATTCTTCTTAACTATTCCCTGATACTCTATGTTTACATATTGCAACAGCCCTCCTGTGCTA
+
BBDE?BB=CB;;A@A?AA<@>2990999=@CE@DBAA<A<G@C499<8<BBGCC?A<959@99919@<94;>@@<
```

Línea del alineamiento (aligned.bed):

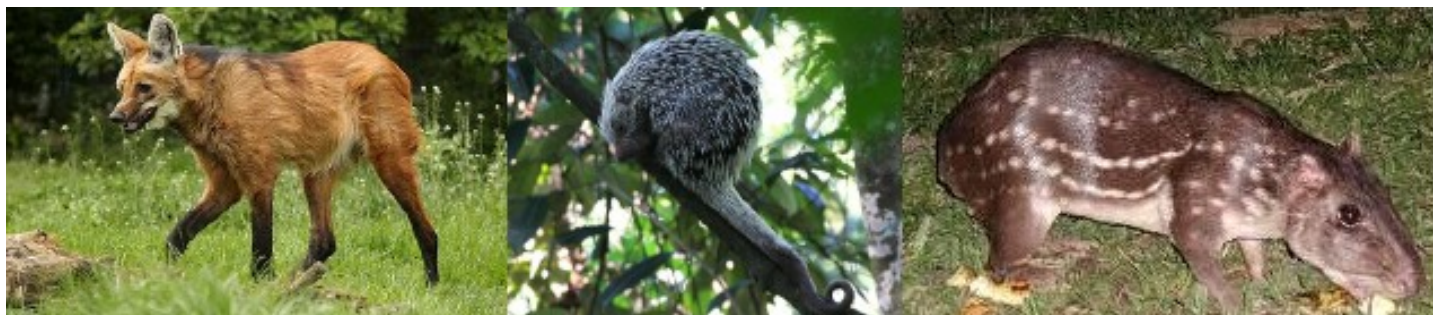
```
gi|576890983|gb|KF964159.1| 106 158 OHYN0:00038:00051 255 +
```

Línea de la base de referencia de d-loop mitocondrial (mam.fasta):

```
>gi|242375783|emb|AM258943.1| Mustela nivalis partial mitochondrial D-loop control
region, tissue library Dk101
ATTCCCTCACCAAATTTTCTATTCATATATTTAACAACATTAATGTGCCTCCCCAGTATGTACTCTTCTT
CCTCCCCCTATGTACTTCGTGCATTAGTGGCTTGCCCCATGCATATAAGCATGTACATATTATGGTTGAT
```

Ejemplo



```
# reads processed: 26451
# reads with at least one reported alignment: 4104 (15.52%)
# reads that failed to align: 22347 (84.48%)
Reported 4104 alignments to 1 output stream(s)
```

SECUENCIAS ALINEADAS

4104 secuencias

15,5%

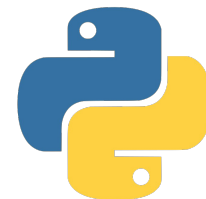
Especie		Lecturas
<i>Cerdocyon thous</i>	zorro de monte	2015
<i>Lycalopex gymnocercus</i>	zorro gris	1085
<i>Conepatus chinga</i>	zorrino	656
<i>Mazama guazoubira</i>	guazubirá	149
<i>Bos taurus</i>	vaca	49
<i>Leopardus wiedii</i>	margay	25
<i>Cuniculus paca</i>	paca	20
<i>Leopardus groffroyi</i>	gato montés	10
<i>Hydrochoerus hydrochaeris</i>	carpincho	8
<i>Sus scrofa</i>	jabalí	6
<i>Canis lupus familiaris</i>	perro	1
INCOSISTENCIAS		80



93,5%

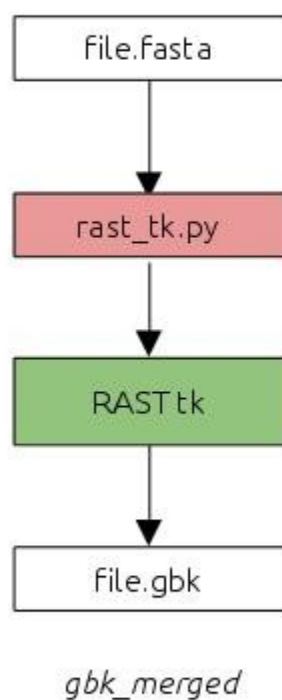


A circular phylogenetic tree showing the relationships between 100 bacterial strains. The tree is color-coded into five main groups: pink (top), orange (top-right), blue (right), green (bottom), and light green (left). The strains are labeled with names like SentM_1, SentM_2, etc., and some are labeled with 'reference' or 'Heidelberg'. The tree shows a high degree of genetic differentiation between the groups, with bootstrap values indicated at the nodes.

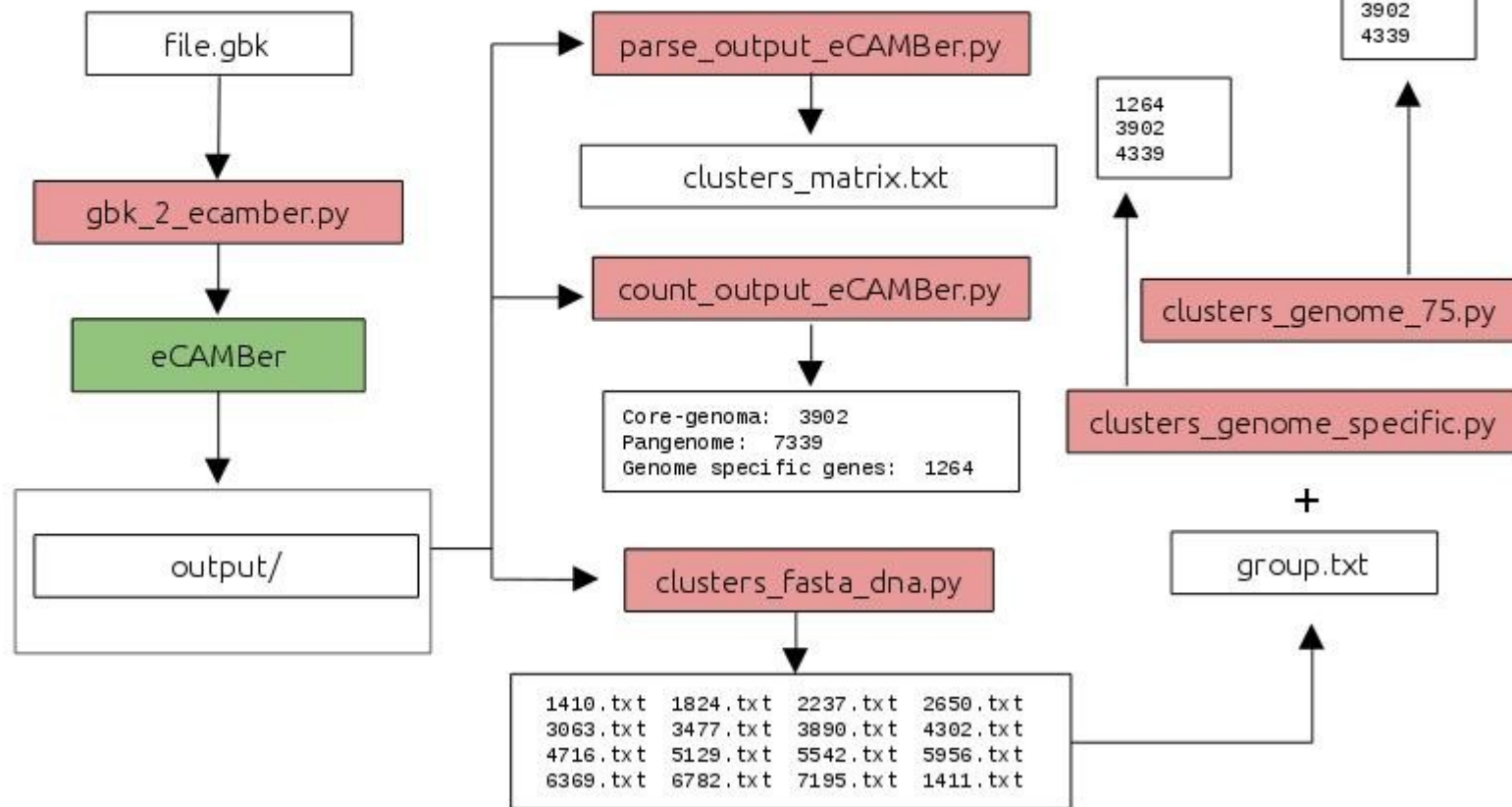


Ejemplo

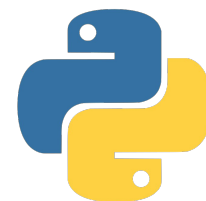
ANOTACIÓN



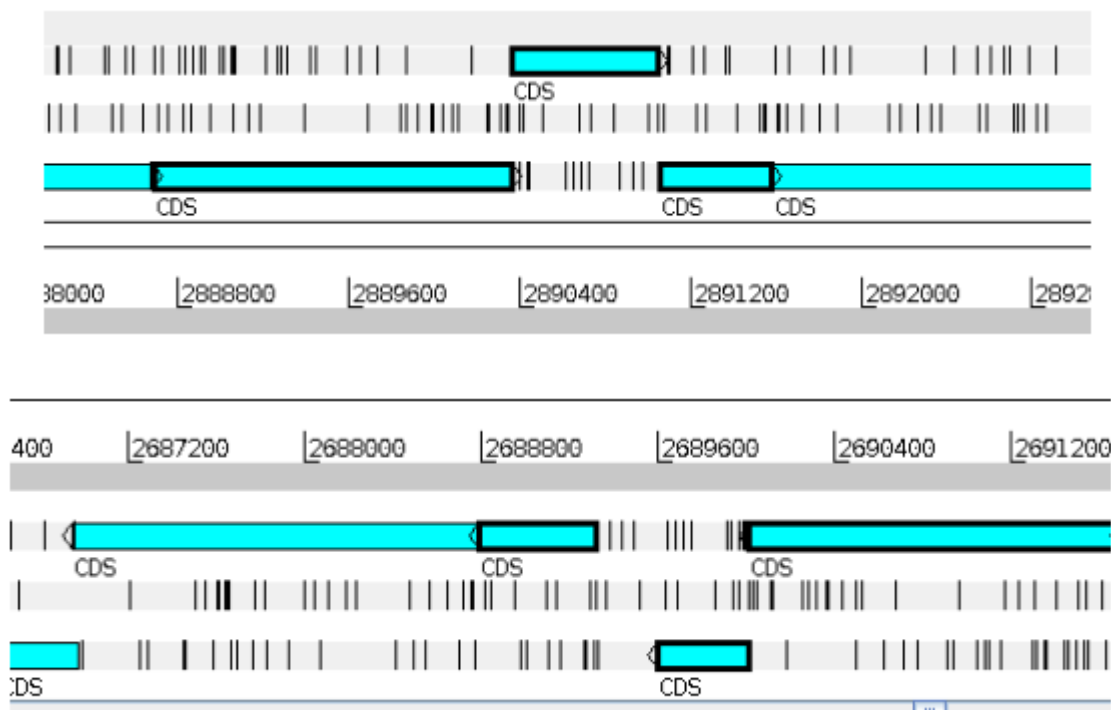
BÚSQUEDA DE ORTÓLOGOS



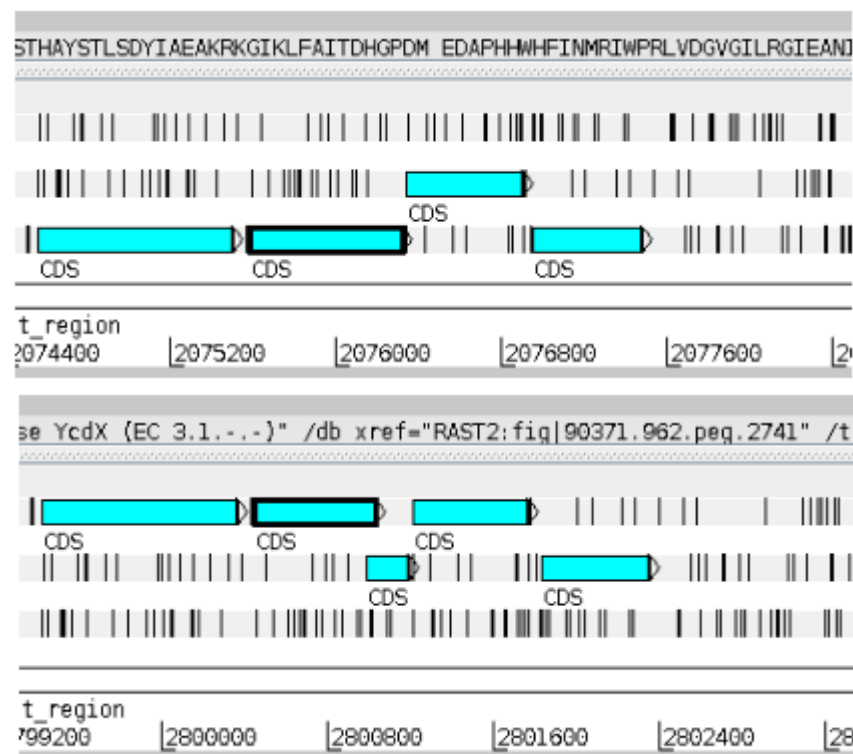
8028 clusters de ortólogos (pangenoma), 4279 contienen familias génicas compartidas por todos los organismos (coregenoma) y 1542 son clusters con genes genoma-específicos



Ejemplo



variante más corta



**introducción de un
codón stop**

Florencia Grattarola
(flograttarola@gmail.com)

