

Taller de

Python

Viernes 6/5, 13/5, 20/5, 27/5
14:30hs Aula 4

Para inscribirte al taller tenes que mandar un mail a cursoslafuente@gmail.com con tu nombre, apellido y DNI o legajo, con el asunto "Inscripción al Taller de Python"

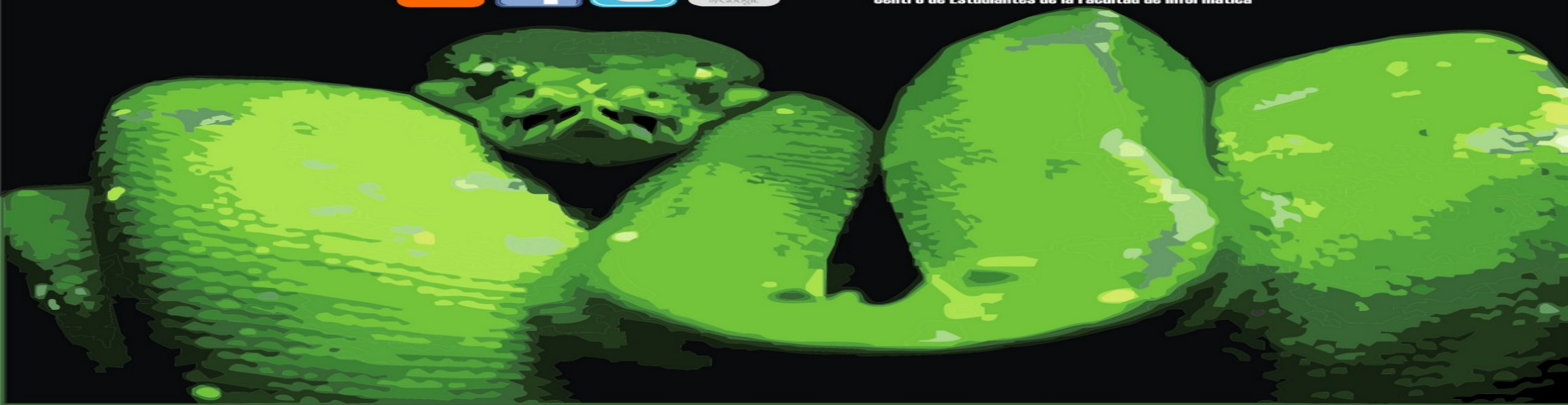
La Fuente



Conducción del

CEFI

Centro de Estudiantes de la Facultad de Informática



¿Que es Software Libre?

- **Software Libre:**

- Se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- **La libertad de usar el programa, con cualquier propósito (libertad 0).**

- **La libertad de estudiar cómo funciona el programa, y adaptarlo a necesidades propias (libertad 1).** El acceso al código fuente es una condición previa para esto.

- **La libertad de distribuir copias (libertad 2).**

- **La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3).** El acceso al código fuente es un requisito previo para esto.

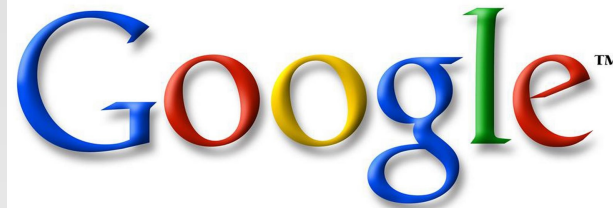
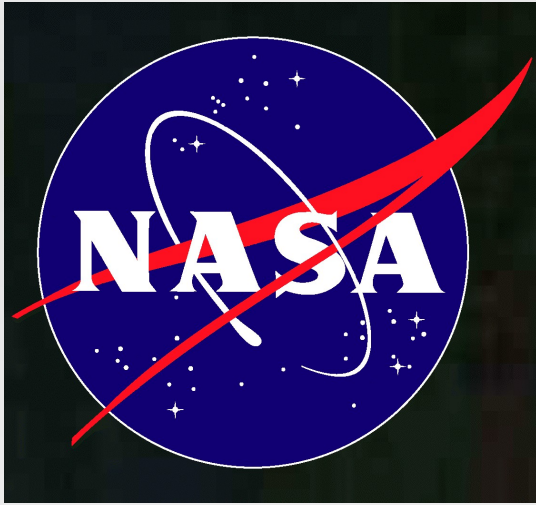
¿Por que programar en python?

Python es un lenguaje de programación de **alto nivel** cuya filosofía hace hincapié en una **sintaxis muy limpia** y que favorezca un código legible.

Se trata de un lenguaje de programación **multiparadigma** ya que soporta orientación a **objetos**, **programación imperativa** y, en menor medida, **programación funcional**. Es un lenguaje interpretado, usa tipado dinámico, es fuertemente tipado y multiplataforma.

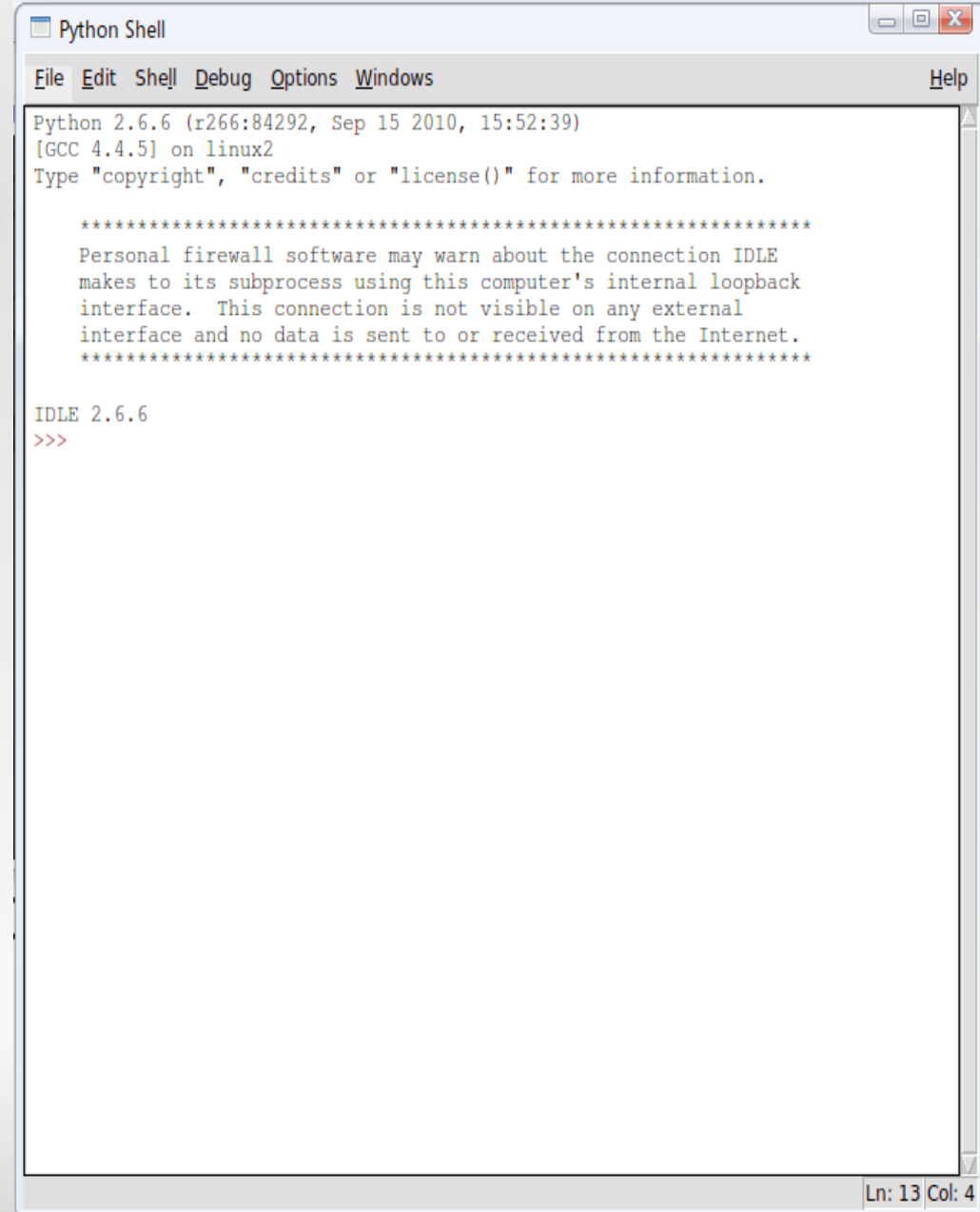


¿Para que sirve?



¿Cómo empezamos?

Existen varias aplicaciones para programar en python. Nosotros en este curso utilizaremos el IDLE.

A screenshot of the Python Shell window. The window has a title bar that says "Python Shell" and a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area displays the following information: "Python 2.6.6 (r266:84292, Sep 15 2010, 15:52:39)", "[GCC 4.4.5] on linux2", and "Type 'copyright', 'credits' or 'license()' for more information." Below this is a warning message about a firewall connection, enclosed in asterisks. At the bottom, it says "IDLE 2.6.6" followed by a prompt ">>>". The status bar at the bottom right shows "Ln: 13 Col: 4".

```
Python Shell
File Edit Shell Debug Options Windows Help
Python 2.6.6 (r266:84292, Sep 15 2010, 15:52:39)
[GCC 4.4.5] on linux2
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 2.6.6
>>>
```

Ln: 13 Col: 4

Formas de ejecutar código

Modo interactivo (sentencia a sentencia)

Modo formal (Escribir el código en un archivo de texto)



Variables y Tipos

- *Variables*

Ejemplo:

- `>>> x = "LALALA"`

- El nombre "x", representa la cadena 'lalala'

- En Python las variables no se declaran.

- El nombre de las variables pueden contener letras, dígitos. (SIEMPRE DEBEN EMPEZAR CON LETRA)

- Importante:

- Hay que asignarle un valor a una variable antes de poder utilizarla.

- Hace diferencia entre mayúsculas y minúsculas: variable x es distinto de variable X



OPERADORES ARITMETICOS

Operador	Descripción	Ejemplo	Resultado
+	Suma	$c = 3 + 5$	$c = 8$
-	Resta	$c = 4 - 2$	$c = 2$
-	Negación	$c = -7$	$c = -7$
*	Multiplicación	$c = 3 * 6$	$c = 18$
**	Potenciación	$c = 2 ** 3$	$c = 8$
/	División	$c = 7.5 / 2$	$c = 3.75$
//	División entera	$c = 7.5 // 2$	$c = 3.0$
%	Módulo	$c = 8 \% 3$	$c = 2$

OPERADORES A NIVEL DE BITS

Operador	Descripción	Ejemplo	Resultado
&	AND	$c = 1 \& 0$	$c = 0$
	OR	$c = 0 1$	$c = 1$
^	XOR	$c = 1 \wedge 1$	$c = 0$
~	NOT	$c = \sim 1$	$c = 0$
<<	Desp. Izquierda	$c = 2 \ll 1$	$c = 4$
>>	Desp. Derecha	$c = 8 \gg 2$	$c = 2$

Tipos de datos básicos

Números:

Enteros, Flotantes y Complejos

>>> var_ent1= 21 Entero

>>> var_ent2= 21L Entero largo

>>> var_ent3 = 027 Octal que representa al número 23 en base 10

>>> var_ent4= 0x17 Hexadecimal que representa al 23 en base 10

>>> var_real1= 0.2703

>>> var_real2= 0.1e-3 Notación científica. Equivale al número: $0.1 \times 10^{-3} = 0.1 \times 0.001 = 0.0001$

>>> i= int(7.5)/2, da como resultado 3

>>> var_comple= 2.1 + 7.8j Tiene su parte imaginaria y su parte real

Operadores Lógicos

operador	comparación
----------	-------------

==	es igual que
!=	es distinto de
<	es menor que
<=	es menor o igual que
>	es mayor que
>=	es mayor o igual que

```
True
>>> 2==3
False
>>> 2==2
True
>>> True == True
True
>>> hola == hola

Traceback (most recent call last):
  File "<pyshell#16>", line 1, in <module>
    hola == hola
NameError: name 'hola' is not defined
>>> 2 == 1+1
True
>>> (True or (2 == 1+2)) == True
True
>>> 2<1
False
>>> 2>2
False
>>> 2>3
False
>>> 3>1
True
>>> -2<=2
True
>>> 1!=1
False
```


Seguimos con cadenas!!!

Operadores de comparación:

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

Ejemplos:

```
>>> 'naranja ' == 'naranja'
```

```
true
```

```
>>> "naranja"<"melon"
```

```
false
```

Python utiliza un criterio de orden alfabético, utiliza los códigos ASCII de los caracteres para decidir su orden.

Para saber el orden que ocupa un carácter se cuenta con las funciones. Predefinidas `ord()` y `chr()`, su función inversa.

```
>>> ord('a')
```

```
97
```

```
>>> chr(78)
```

```
'N'
```



Funciones para Cadenas

Función	Descripción	Ejemplo
<code>int()</code>	Convierte la cadena numérica a entero	<pre>>>> int("123") 123</pre>
<code>float()</code>	Convierte la cadena numérico a flotante	<pre>>>> float("123") 123.0</pre>
<code>str()</code>	Convierte un número a string	<pre>>>> str(23) '23'</pre>
<code>ord()</code>	Devuelve el código ASCII (número entero) correspondiente del carácter	<pre>>>> ord("a") 97</pre>
<code>chr()</code>	Devuelve el carácter correspondiente al número	<pre>>>> chr(89) 'y'</pre>

–Algunos métodos **propios** de las cadenas:

Métodos	Descripción	Ejemplo
<code>a.lower()</code>	Convierte los caracteres de la cadena "a" a minúscula	<pre>>>> pal='HOLA' >>> print pal.lower() hola</pre>
<code>a.upper()</code>	Convierte los caracteres de la cadena "a" a mayúscula	<pre>>>> pal='hola' >>> print pal.upper() HOLA</pre>

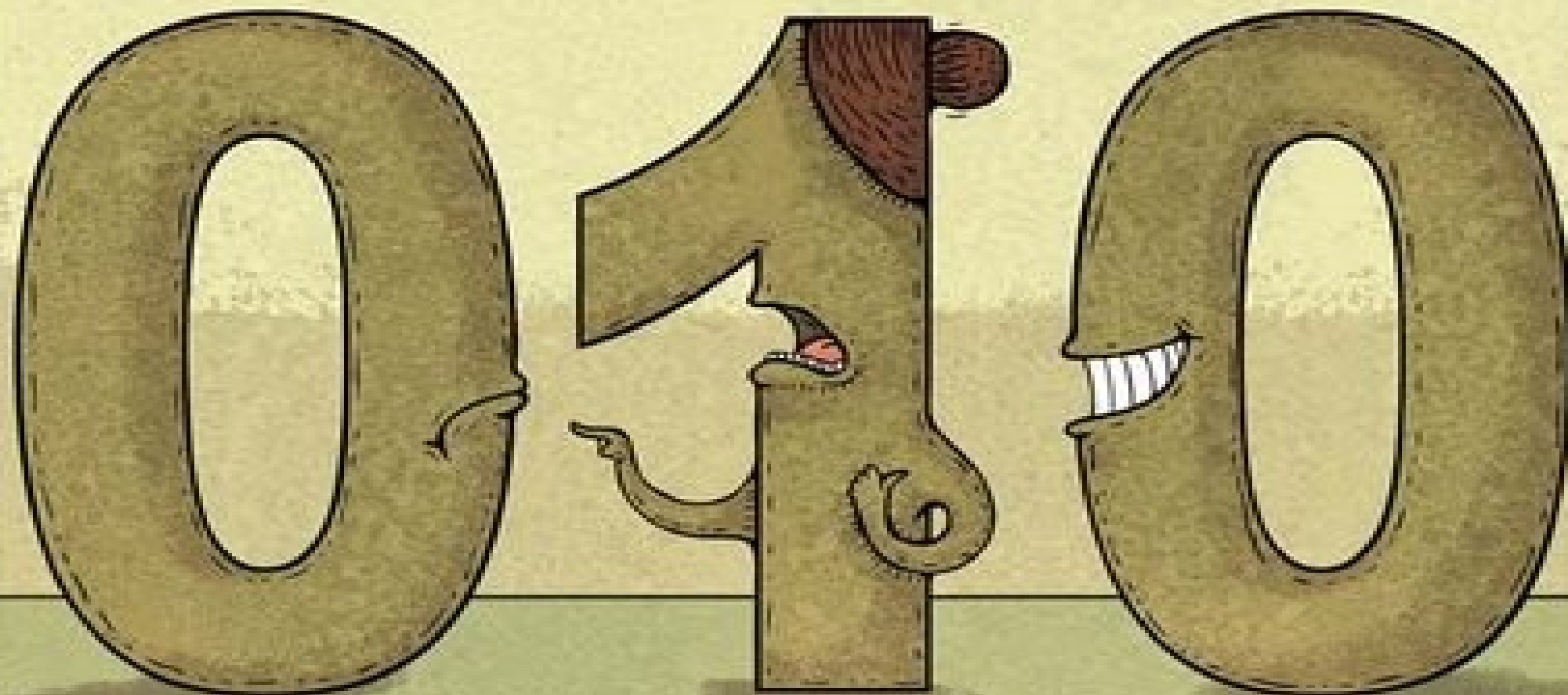
•Cadenas (cont.): Scape

Secuencia de escape para carácter de control	Resultado
<code>\a</code>	Carácter de «campana» (BEL)
<code>\b</code>	«Espacio atrás» (BS)
<code>\f</code>	Alimentación de formulario (FF)
<code>\n</code>	Salto de línea (LF)
<code>\r</code>	Retorno de carro (CR)
<code>\t</code>	Tabulador horizontal (TAB)
<code>\v</code>	Tabulador vertical (VT)
<code>\ooo</code>	Carácter cuyo código ASCII en octal es <i>ooo</i>
<code>\xhh</code>	Carácter cuyo código ASCII en hexadecimal es <i>hh</i>

¿Se entendió?

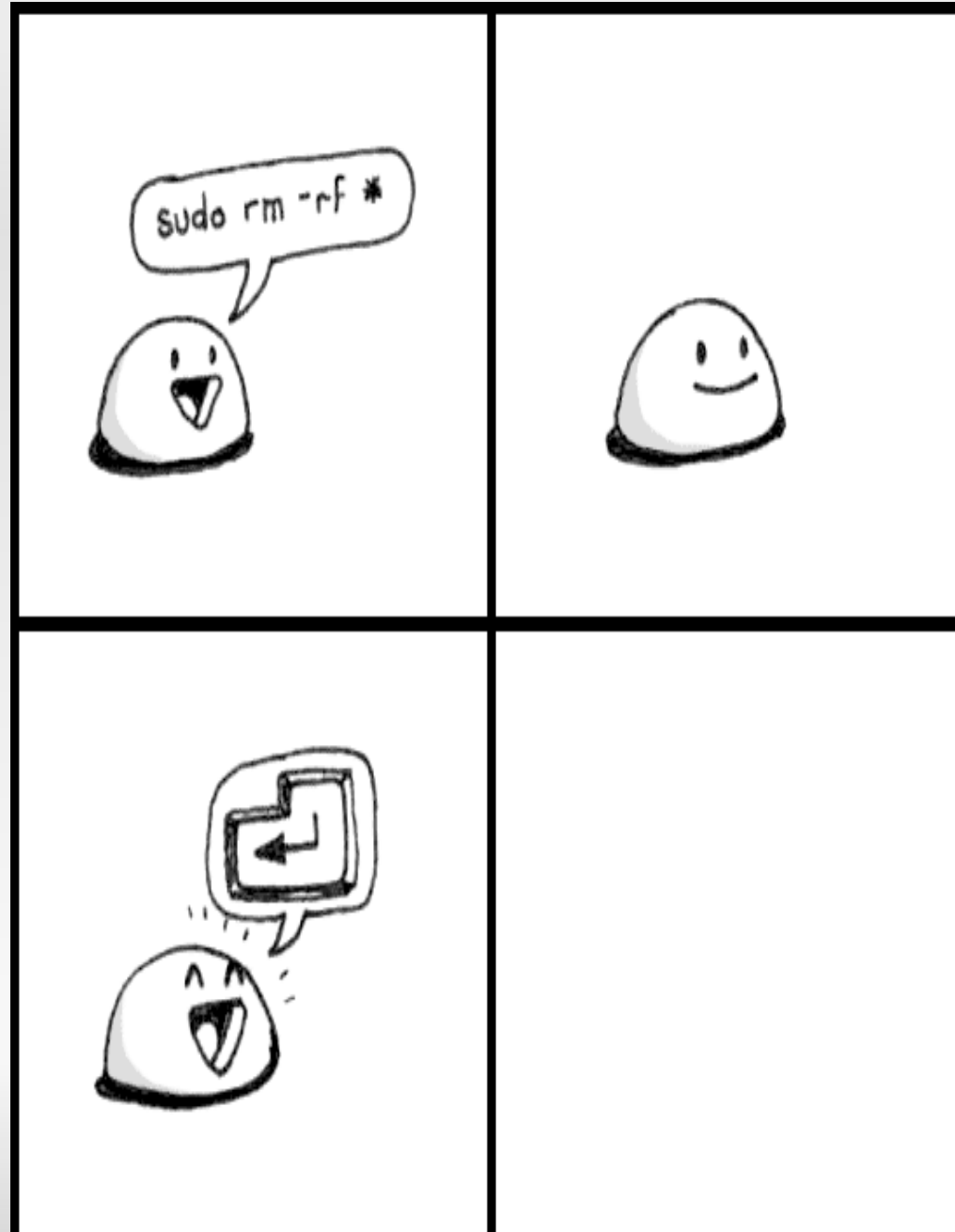
SIEMPRE FUISTE UN INÚTIL, JAVIERITO.

¡NO SÉ POR QUÉ NO PUEDES SER UN POCO MÁS PARECIDO A TU HERMANO!



Algunas funciones más de cad.

- `len("")` devuelve longitud 0
- `len('B')` devuelve longitud 1
- `len(' ')` devuelve longitud 1
- `len('BBS')` devuelve longitud 3



Slicing

El operador : (slicing), nos permite obtener subcadenas.

[:] devuelve toda la cadena

Indices negativos, recorren de derecha a izquierda la cadena

```
>>> cadena_python = 'Me quiero ir estoy aburrido'
>>> print cadena_python [0]
M
>>> print cadena_python [3:8]
quier
>>> print cadena_python [10:12]
ir
>>> print cadena_python [8:-1]
o ir estoy aburrid
```



LISTAS!

Colección ordenada, equivalente a Arrays o Vectores

–Puede contener cualquier tipo de datos, **inclusive listas**.

Ej.: `lis1= [42, False, 'Curso python', [1,4]]`

–**Forma de acceder a sus elementos:**

- Indicar el **índice** del elemento (posición dentro de la lista), entre corchetes []. **IMPORTANTE:** los índices comienzan en 0.

Ej.: `lis1[1] = True`, esto provoca que el 2do elemento de la lista se cambie al valor verdadero.

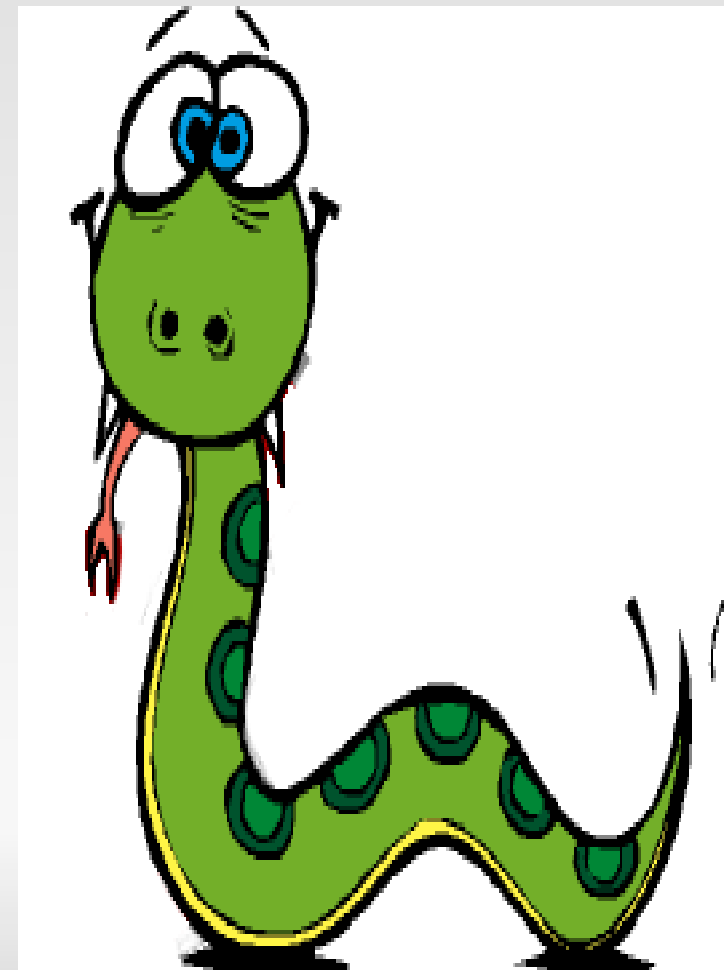
- Para acceder a elementos “listas”, se debe usar también []. El primero indica posición de la lista exterior, los otros indican posición de las listas interiores.

Ej.: `lis1[3][1]`, devuelve 1

- Se pueden usar **índices negativos**. En ese caso se **comienza a contar desde atrás**.

Ej.: `lis1[-3]`, devuelve False

EL Slicing TAMBIEN SE APLICA A LISTAS



Unmotivating.com



GAME OVER