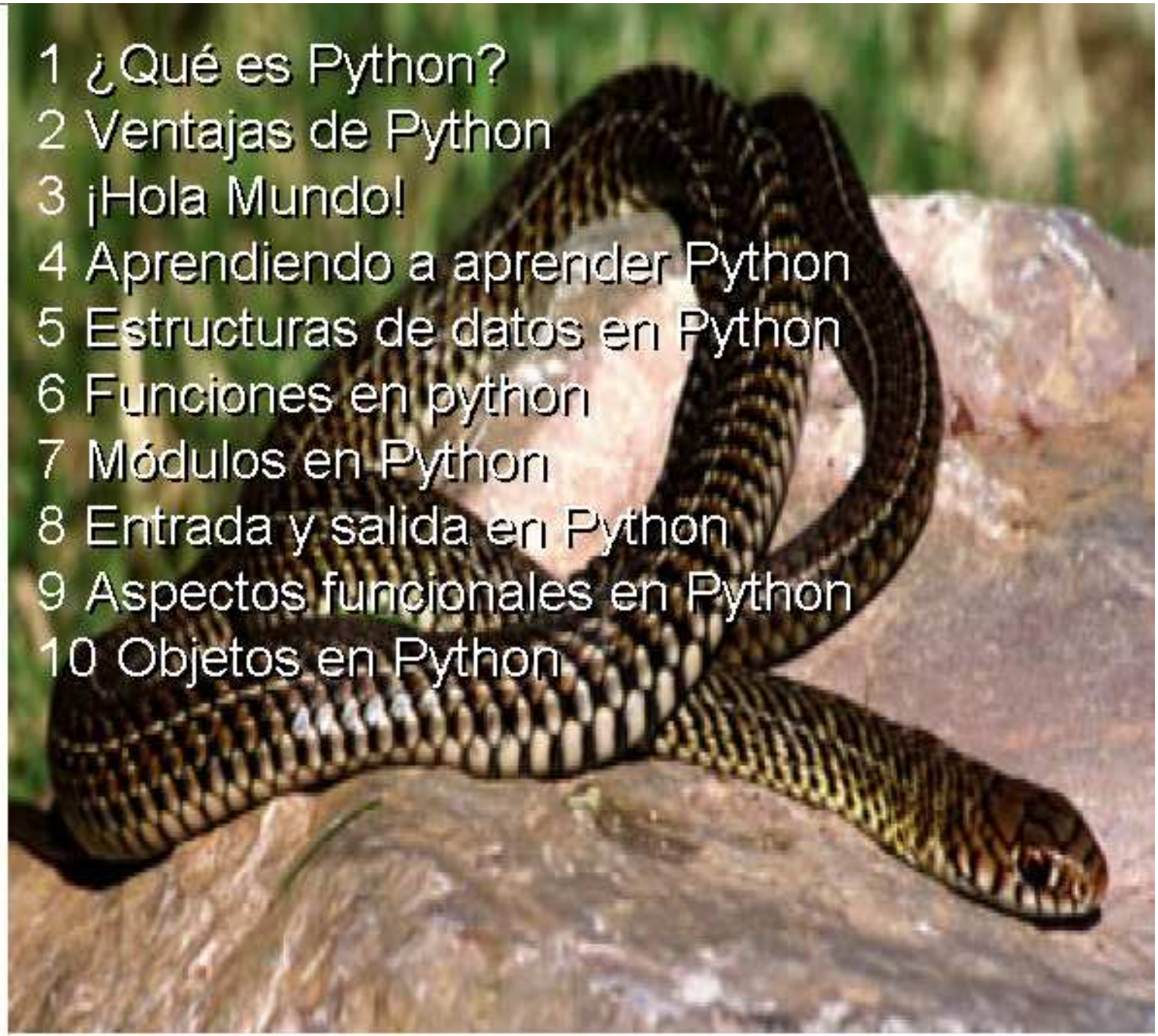
Introducción al lenguaje de programación Python

José Millán Soto fid@gpul.org
http://fid.homelinux.org/gpul

ÍNDICE



¿Qué es Python?

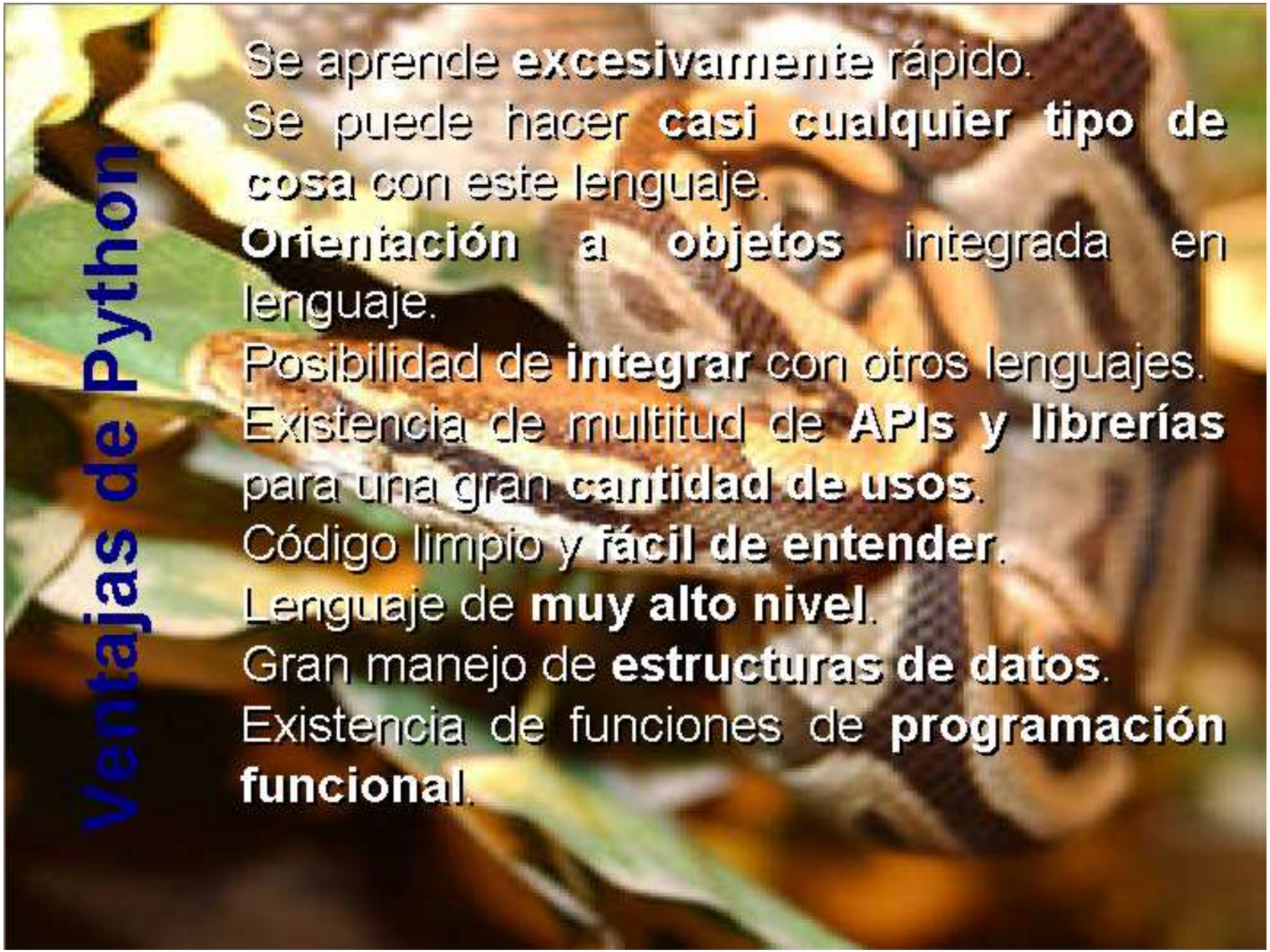
a **Serpiente pitón:** Nombre común de la familia Boidae de serpientes no venenosas, específicamente la subfamilia Pythonidae.
b **Mitología griega:** Gran serpiente, nacida del barro que quedó en la tierra después del gran diluvio. El monstruo vivía en una gruta cerca de Delfos y allí custodiaba el oráculo. El dios Apolo mató a la Pitón, fue conocido como Apolo Pitio.
c **Monty Python:** Grupo de humoristas creadores del programa televisivo "Monty Python's Flying Circus" y cuatro películas.
El nombre del lenguaje está inspirado en uno de los tres nombres, ¿En cuál?

LA RESPUESTA C

¿Qué es Python?

Python es un lenguaje de programación semi-interpretado, semi-compilado.
El lenguaje Python es creado por **Guido Van Rossum**.
Está basado en **ABC**.
Es un lenguaje muy fácil de **aprender**, además de **extensible y potente**.
Es un lenguaje de programación de **muy alto nivel** y con un **gran nivel de abstracción**.
Cuando termine esta charla además sera:

¡¡Vuestro lenguaje de programación favorito!!


Ventajas de Python
Se aprende excesivamente rápido.
Se puede hacer casi cualquier tipo de cosa con este lenguaje.
Orientación a objetos integrada en lenguaje.
Posibilidad de integrar con otros lenguajes.
Existencia de multitud de APIs y librerías para una gran cantidad de usos.
Código limpio y fácil de entender.
Lenguaje de muy alto nivel.
Gran manejo de estructuras de datos.
Existencia de funciones de programación funcional.

¡Hola mundo!

```
#!/usr/bin/env python
print 'Hola mundo!!!!'
```

El trozo de código de ahí arriba es un ejemplo de programa en python.
Otro código python:

```
#!/usr/bin/env python
print '¿Cómo te llamas?'
nombre = raw_input()
print "Hola", nombre
```

Aprendiendo a aprender Python

Python es un lenguaje muy simple de aprender. Veamos su sintaxis básica:

Comentarios: #Esto es un comentario

Asignación:

```
n = 42
k = p = 0
m, n = 20, 40
k -= 1   #Ahora k es igual a -1
```

Condicional:

```
if (n < 250): print "n es pequeño"
elif (n==250): print "n es grande"
else: print "n es muy grande"
```

Iteraciones:

```
for i in range(25): print i
```

Bucles:

```
while (i < 2): i+= 1
```

Aprendiendo a aprender Python

Instruccción pass: No hace nada, debe usarse cuando es requerida sintácticamente.

Bloques de código:

Para utilizar bloques de código se indenta el código:

```
print "¿Como te llamas?"
nombre = raw_input()
if len(nombre) == 0:
    print "No me has respondido"
    print "Me parece mal"
else:
    print "Hola", nombre
```

Al utilizar código indentado como bloques de código, el código siempre es muy limpio.

Aprendiendo a aprender Python

Uso del intérprete en modo interactivo:

- Iniciando el comando python sin argumentos se inicia el intérprete en modo interactivo
- El intérprete en modo interactivo **soporta readline**
  - No en todos los SOs :(
- Funciones útiles: **dir**, **help**

Estructuras de datos en Python

Python dispone de estructuras de datos implementadas por defecto.

- Listas: Estructura de datos formada por una sucesión de datos. **Sus elementos pueden variar.** Ej: [3, 'Hola']
- Tupla: Estructura de datos formada por una sucesión de datos. **No mutable**. Ej: (3, 'Hola')
- Diccionario: **Estructura de datos formada por datos e identificadores**. Sus elementos pueden variar. No pueden usarse listas como identificadores. Ej: { 'n': 3, 4: 'Hola'}

Estructuras de datos en Python

Trabajando con estructuras de datos

Veamos en un ejemplo como trabajar con estas estructuras

Funciones en Python

Para definir funciones se utiliza la siguiente estructura:

```
def suma(n, x):
  return n + x

def sumar(n, x):
  resultado = suma(n, x)
  return resultado
```

Una función es un dato más, en vez de hacer la definición anterior, podíamos haber hecho:

```
sumar = suma
```

Módulos en python

Python es un lenguaje muy modulable.
Existen librerías para todo
Estructuras import y from - import
Veamoslo con un ejemplo

Entrada y salida en python

Entrada y salida estándar:

- Para mostrar una línea en pantalla:
  print 'Soy un texto de prueba que mola'
- Para mostrar un texto en pantalla:
  print 'No avanza linea',
  OJO: No avanza línea, pero escribe un espacio.
- Para leer una línea de pantalla:
  info = raw_input()
- Archivos para entrada y salida estándar: (necesario importar sys)
  - **sys.stdin**
  - **sys.stdout**
  - **sys.stderr**

Entrada y salida en python

Entrada y salida con archivos:

- Para abrir un archivo:
  Instrucción open
- Funciones de lectura y escritura de archivo:
  eFile = open('archivo','rb+')
  eFile.read()
  eFile.write()
  eFile.seek()
- Otros formatos de entrada y salida:
  Pipes: os.popen()
  También existen popen2 y popen3
  Sockets

Veamos un ejemplo

Entrada y salida en python

Entrada y salida avanzada:

- Guardar datos con tipos en archivos:
  - Módulo pickle
  - Módulo marshal

Entrada y salida en python

Funciones de string:

```
>>> cad='Hola'
>>> cad.rjust(10)
'      Hola'
>>> cad.ljust(10)
'Hola      '
>>> cad.ljust(3)
'Hola'
>>> `9`
'9'
>>> `(8, True)`
'(8, True)'
>>> `8`.zfill(4)
'0008'
```

Programación funcional en Python

Python dispone de unas estructuras para la programación funcional:

- **Funciones** de programación funcional:
  - map
  - filter
  - reduce

- **Funciones lambda**:
  - Palabra clave lambda:

  Permite generar funciones anónimas

VEAMOS UNOS EJEMPLOS

Programación funcional en Python

Ejemplos:

```
>>> map((lambda x: x+1),[3, 4])
[4, 5]
>>> def genSuma(num):
...   return (lambda x,m=num : x+m)
...
>>> genSuma(9)(10)
19
>>> def menor(num):
...   return (lambda x,m=num: x<m)
...
>>> filter(menor(9),range(42))
[0, 1, 2, 3, 4, 5, 6, 7, 8]
>>> reduce(genSuma(2),range(10))
45
```

Orientación a objetos en Python

Python es un lenguaje orientado a objetos.
En Python todo es un objeto:
Ejemplo de definición de clase:

```
class Animal:
    """Clase que representa un animal
    """
    def hacerRuido(self):
        """Hace el ruido propio del animal al
que representa
        """
        print 'No se que ruido hago'
class Perro(Animal):
    """Los perros ladran
    """
    def __init__(self, nombre):
        self._nombre=nombre
    def hacerRuido(self):
        """Dice Guau Guau
        """
        print self._nombre+':','Guau Guau'
```

Recursos interesantes:

- http://python.org
- Tutorial de python de Guido Van Rossum
- Begginer's Guide to Python (python.org)
- Charlas sobre programación de interfaces gráficas en Python, las dos siguientes ;)


G.P.U.I.
Turno de preguntas y/o comentarios


G.P.U.I.
Muchas gracias por venir.