



PYTHON

Desde lo informal al lenguaje

Hemos aprendido cómo describir una tarea básica como un conjunto de instrucciones simples.



Llegó la hora de expresar esos algoritmos en un lenguaje que pueda manipularse en el computador.

Aprendimos también como expresar el algoritmo en un lenguaje formal aunque de naturaleza humana.





¿Qué es un lenguaje de
programación?

PYTHON!



- Es una versión informática de un "pseudo-lenguaje"
- Tiene las mismas componentes y sigue también ciertas reglas (sintaxis).
- Están diseñados con reglas muy estrictas y restricciones relacionadas con la naturaleza del computador.



¿Cuáles son los tipos de
lenguaje de programación?

PYTHON!





¿Qué tipo de lenguaje es
Python?

PYTHON!



Lenguaje de Alto nivel.



Lenguaje interpretado (scripts).



Software Libre (GPL).



Multiplataforma. [Anaconda](#) + [VScode](#)



Programación orientada a objetos.



Puntos fuertes: Gráficas, Machine learning, Instrumental, Software científico, Comunidad.



Puntos debiles: Programación del alto performance (uso de Wrappers)





PYTHON

Resolviendo problemas usando la consola
y scripts

USANDO PYTHON EN CONSOLA

```
[usuario@maquina]$ python
Python 3.6.4 |Anaconda custom (64-bit)| (default, Jan 16
    2018, 18:10:19)
[GCC 7.2.0] on linux
Type "help", "copyright", "credits" or "license" for
    more information.
>>>
```

Ayuda del interprete:

```
>>> help()
help> math
help> q
```

USANDO PYTHON EN CONSOLA

```
>>> a = 5 + 3
```

```
>>> a
```

⇒ Operaciones
básicas

```
>>> cos(90)
```

⇒ Operaciones
matemáticas

```
>>> import math
```

```
>>> math.cos(90)
```

⇒ Limpiar
pantalla y
salir

```
>>> import os
```

```
>>> os.system('clear')
```

```
>>> exit()
```

VARIABLES EN PYTHON

En Python las variables son “tipeadas” dinámicamente.

```
>>> a=3
>>> type(a)
>>> a=3.1
>>> type(a)
>>> b=2
>>> var=a+b
>>> type(var) # tipo de var
>>> var=int(a)+b # pasar "a" a entero
>>> print(var)
```

`bool()`

`complex()`

`float()`

`str()`

`int()`



TEN CUIDADO CON EL
“TIPEO” DINÁMICO

MI PRIMER SCRIPT EN PYTHON



Importar el módulo math al principio.



La función Leer es reemplazada por la función **input**.



La función Imprimir/Escribir es reemplazada por la función **print**.



Las funciones integradas son [Built-in Functions](#)



Las funciones del módulo math son [Math functions](#)



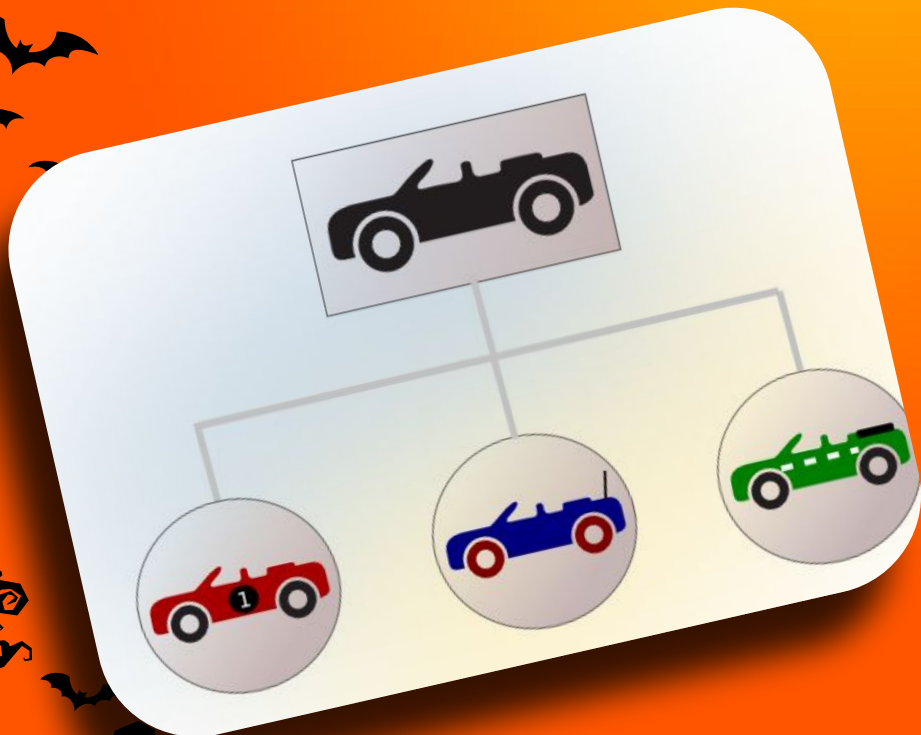
El script debe recibir un número y determinar si es un entero o no.



OBJETOS EN PYTHON

Resolviendo problemas usando la consola
y scripts

TODO EN PYTHON ES UN OBJETO



Las clases son plantillas que definen a un objeto en concreto.

Objeto es un instancia de una clase.

TODO EN PYTHON ES UN OBJETO

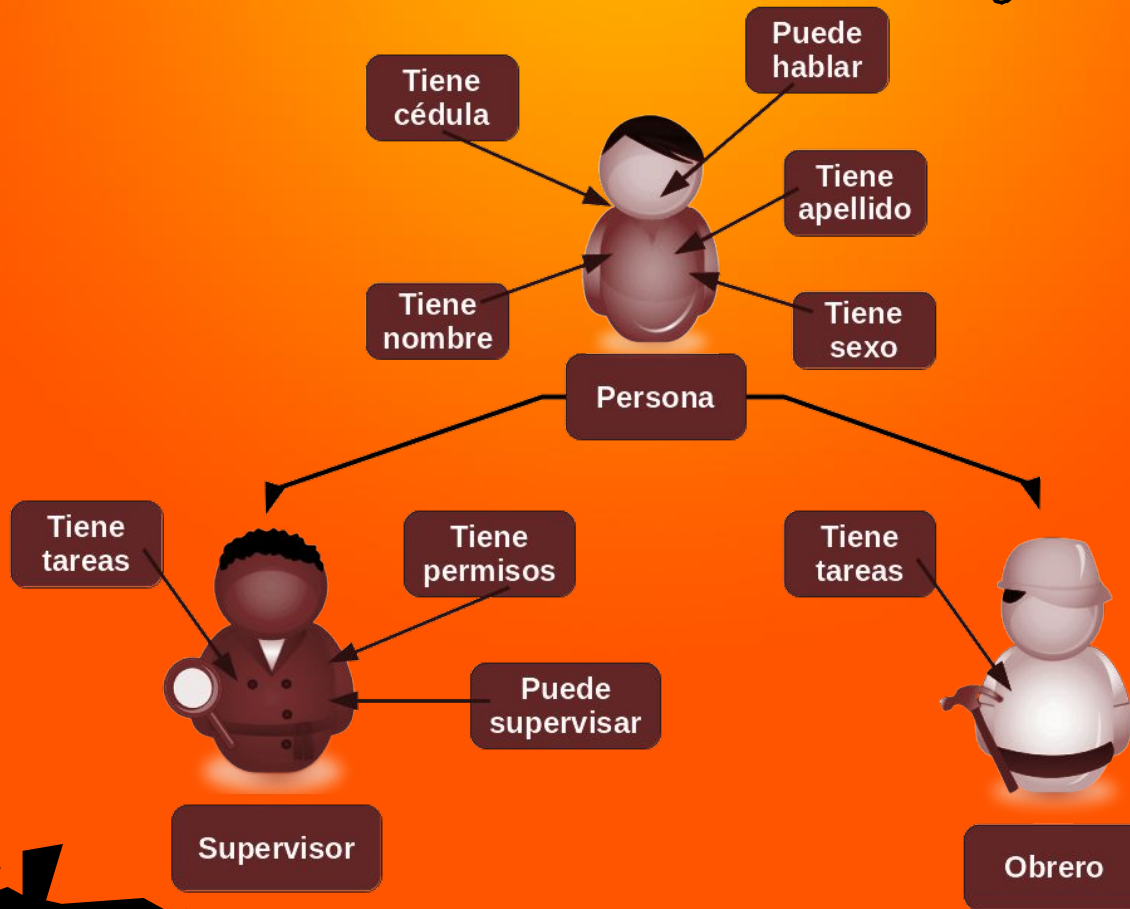
Los objetos tienen atributos (propiedades)

Los objetos pueden realizar acciones (métodos)

Los objetos pueden contener otros atributos-objetos (composición)



TODO EN PYTHON ES UN OBJETO



Prueba en una consola usando el
interprete de python3:

```
a=2.5
```

```
a.is_integer()
```

```
a.as_integer_ratio()
```

The background of the image is a close-up photograph of several large, ripe orange pumpkins. The pumpkins have prominent ridges and green stems. Scattered throughout the image are white, stylized Halloween-themed icons: small stars, flying bats, and decorative trees. At the bottom of the image, there is a white silhouette of a landscape featuring a witch's hat, a broom, a cat, and a jack-o'-lantern.

HAY MÁS CLASES EN
PYTHON

LAS CADENAS DE CARACTERES

```
>>> string1 = "hola"  
>>> string2 = 'mundo'  
>>> print(string1 + " " + string2)  
hola mundo  
>>> print(string1[2:])  
la  
>>> string2[2] = "b"
```

- ★ Las cadenas son secuencia de caracteres
- ★ Pueden concatenarse con el operador suma (+).
- ★ Con el operador (:) se puede extraer una subcadena.
- ★ Son objetos inmutables.

LAS CADENAS DE CARACTERES

- ★ `str.capitalize()`: Primera letra en mayúscula.
- ★ `str.lower()`: Todas las letras en minúscula.
- ★ `str.upper()`: Todas las letras en mayúscula.
- ★ `str.swapcase()`: Invertir mayúsculas y minúsculas.
- ★ `str.title()`: Cada letra en mayúscula.
- ★ `str.center(50[, symbol])` Centrar cadena sola o con un símbolo.
- ★ `str.count(carácter)`: Permite contar cuántas veces está el carácter en la cadena.
- ★ `str.find(str)`: Permite buscar una subcadena en una cadena.

LAS CADENAS DE CARACTERES

- ★ `str.isalnum()` Saber si es alfanumérica.
- ★ `str.isalpha()` Saber si es alfabética.
- ★ `str.isdigit()` Saber si es numérica.
- ★ `str.islower()` Saber si sólo tiene minúsculas.
- ★ `str.isspace()` Saber si tiene espacios.
- ★ `str.replace(cadena buscar, cadena a reemplazar)`
Reemplaza una cadena por otra (no in situ).
- ★ `str.strip()`: Eliminar caracteres a la derecha e izquierda de una cadena.
- ★ `str.split([separador])`: Separa una cadena en varias partes según separador.

MI SEGUNDO SCRIPT EN PYTHON

Crear un script que debe recibir una contraseña (sólo números y letras) y me indique si

1. Tiene espacios u otro carácter.
2. Tiene más de 8 caracteres.
3. Tiene por lo menos una Mayúscula (si sólo tiene números y letras) (Negación es not)
4. Tiene por lo menos un número (si sólo tiene números y letras).

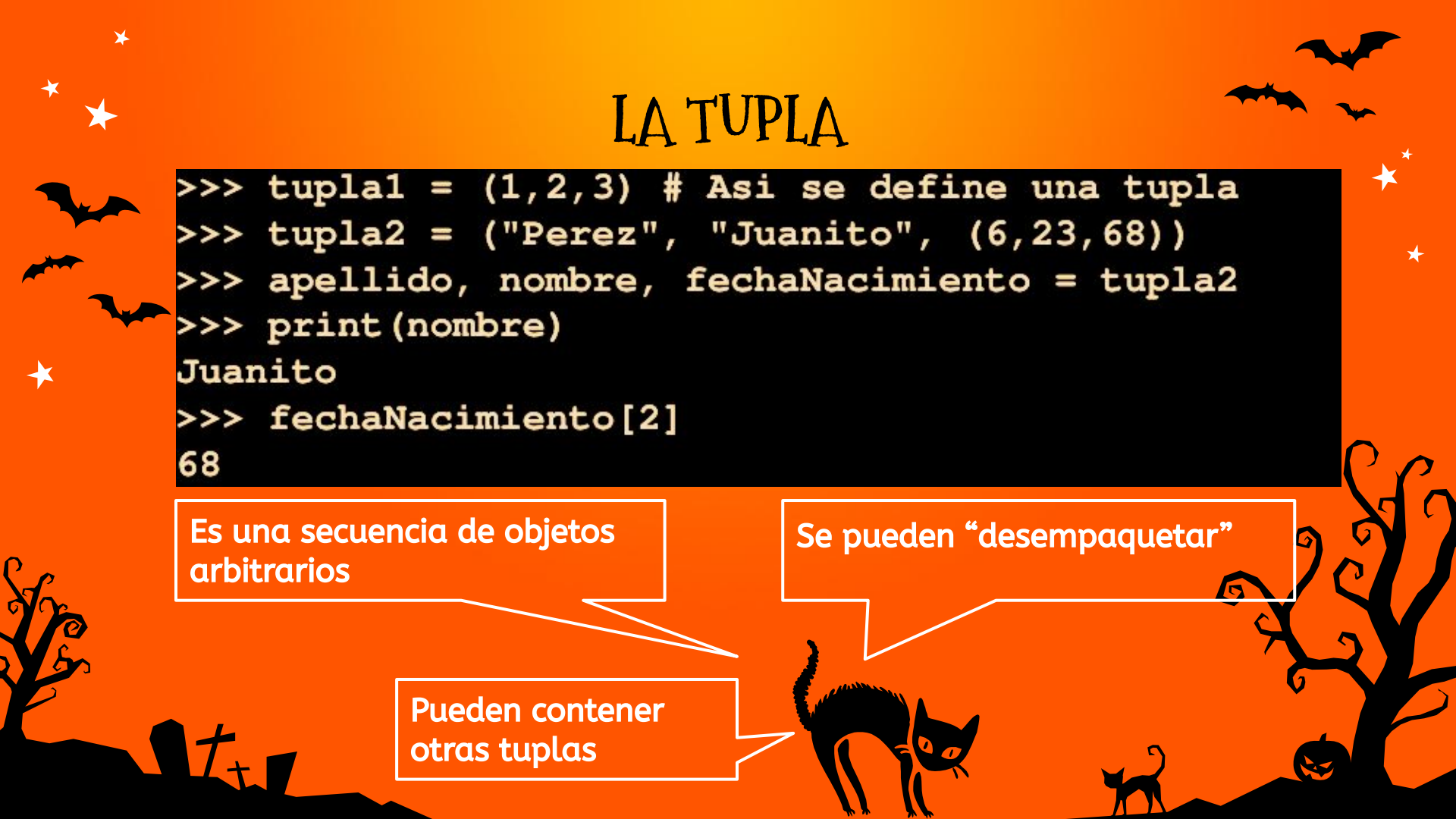
LA TUPLA

```
>>> tupla1 = (1,2,3) # Asi se define una tupla
>>> tupla2 = ("Perez", "Juanito", (6,23,68))
>>> apellido, nombre, fechaNacimiento = tupla2
>>> print(nombre)
Juanito
>>> fechaNacimiento[2]
68
```

Es una secuencia de objetos
arbitrarios

Se pueden “desempaquetar”

Pueden contener
otras tuplas



LA TUPLA

```
>>> nombreCompleto = tupla2[1] + " " + tupla2[0]
>>> print(nombreCompleto)
Juanito Perez
>>> print(tupla2[0:2])
('Perez', 'Juanito')
```

Son inmutables.

tuple.count()
tuple.index()

Soportan operaciones
similares a los strings.



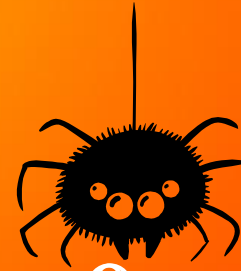
LA LISTA

```
>>> a = [1.0, 2.0, 3.0] # Creando una lista
>>> a.append(4.0) # Agregando 4.0 a una lista
>>> print(a)
[1.0, 2.0, 3.0, 4.0]
>>> a.insert(0,0.0) # Insertando 0.0 en la pos. 0
>>> print(a)
[0.0, 1.0, 2.0, 3.0, 4.0]
>>> len(a) # Determina la longitud de una lista
5
>>> a[2:4] = [1.0, 1.0] # Modificando elementos
```

Es como una tupla
pero mutable

LA LISTA

```
>>> a = [1.0, 2.0, 3.0]
>>> b = a
>>> b[0] = 0.5
>>> print b
>>> print a
>>> c = a[:]
>>> c[0] = 1.0
>>> print a
```



¡Las listas
son
punteros!

LA LISTA

- ★ `list.append(x)`: Agrega un ítem al final de la lista;
- ★ `list.extend(L)`: Extiende la lista agregándole todos los ítems de la lista dada
- ★ `list.insert(i, x)`: Inserta un ítem en una posición dada. El primer argumento es el índice del ítem delante del cual se insertará, por
- ★ `list.remove(x)`: Quita el primer ítem de la lista cuyo valor sea x. Es un error si no existe tal ítem.
- ★ `list.count(x)`: Devuelve el número de veces que x aparece en la lista.
- ★ `list.sort()`: Ordena los ítems de la lista, in situ.
- ★ `list.reverse()`: Invierte los elementos de la lista, in situ.



LA LISTA

- ★ `list.pop([i])`: Quita el ítem en la posición dada de la lista, y lo devuelve. Si no se especifica un índice, `a.pop()` quita y devuelve el último ítem de la lista.
- ★ `list.index(x[, start[, end]])`: Devuelve el índice en la lista del primer ítem cuyo valor sea `x`. Es un error si no existe tal ítem. Los argumentos opcionales `start` y `end` son interpretados como la notación de rebanadas y se usan para limitar la búsqueda a una subsecuencia particular de la lista.
- ★ `list.copy()`: Devuelve una copia superficial de la lista.
- ★ `list.clear()`: Quita todos los elementos de la lista.



MI TERCER SCRIPT EN PYTHON

Crear un script que recibe 5 números enteros y:

1. Los guarda en una lista llamada entradas.
2. Los organiza de mayor a menor en un lista llamada orden.
3. Revisa uno por uno cuántas veces se repite el número.


EL DICCIONARIO

```
>>> eng2sp = {} # Definiendo un diccionario vacio
>>> eng2sp['one'] = 'uno'
>>> eng2sp['two'] = 'dos'
>>> print(eng2sp)
{'two': 'dos', 'one': 'uno'}
>>> eng2sp = {'one': 'uno', 'two': 'dos', 'three':
'tres'}
>>> print(eng2sp)
{'three': 'tres', 'two': 'dos', 'one': 'uno'}
```

Son un tipo de estructuras de datos que permite guardar un conjunto no ordenado de pares clave-valor

EL DICCIONARIO

- ★ `dict.clear()`: Quita todos los elementos del diccionario.
- ★ `dict.copy()`: Devuelve una copia superficial del diccionario.
- ★ `dict=dict.fromkeys(list,valor)`: Crea un diccionario desde una lista de llaves o claves con un valor por defecto.
- ★ `dict.setdefault('clave' [, None|valor_por_defecto])` Si la clave no existe, la crea con el valor por defecto. Siempre retorna el valor para la clave pasada como parámetro.
- ★ `dict.get(clave[, "valor x defecto si la clave no existe"])`: Obtener el valor de una clave
- ★ `dict.items()` Obtener las claves y valores de un diccionario
- ★ `dict.keys()`: Obtener las claves de un diccionario.
- ★ `dict.values()`: Obtener los valores de un diccionario.



ESTRUCTURAS EN PYTHON

Resolviendo problemas usando la consola
y scripts

LA ESTRUCTURA CONDICIONAL

```
numero =float(input("Introduzca un numero "))  
if numero>0:  
    print ("El número es mayor que 0")  
elif numero<0:  
    print ("El número es menor que 0")  
else:  
    print ("El número es igual a 0")
```


★ LOS CICLOS WHILE (MIENTRAS) Y FOR (PARA)

```
import math

nMax = 5
n = 1
a = []
while n < nMax:
    a.append(1.0/n)
    n = n + 1
print(a)
```

```
import math

nMax = 5
a = []
for n in range(1, nMax):
    a.append(1.0/n)
print(a)
```



SyntaxError:
invalid syntax

IndentationError
: expected an
indented block

NO SE TE OLVIDEN LOS (:)
Y LOS ESPACIOS (TAB)

LAS SENTENCIAS DE CONTROL EN CICLOS

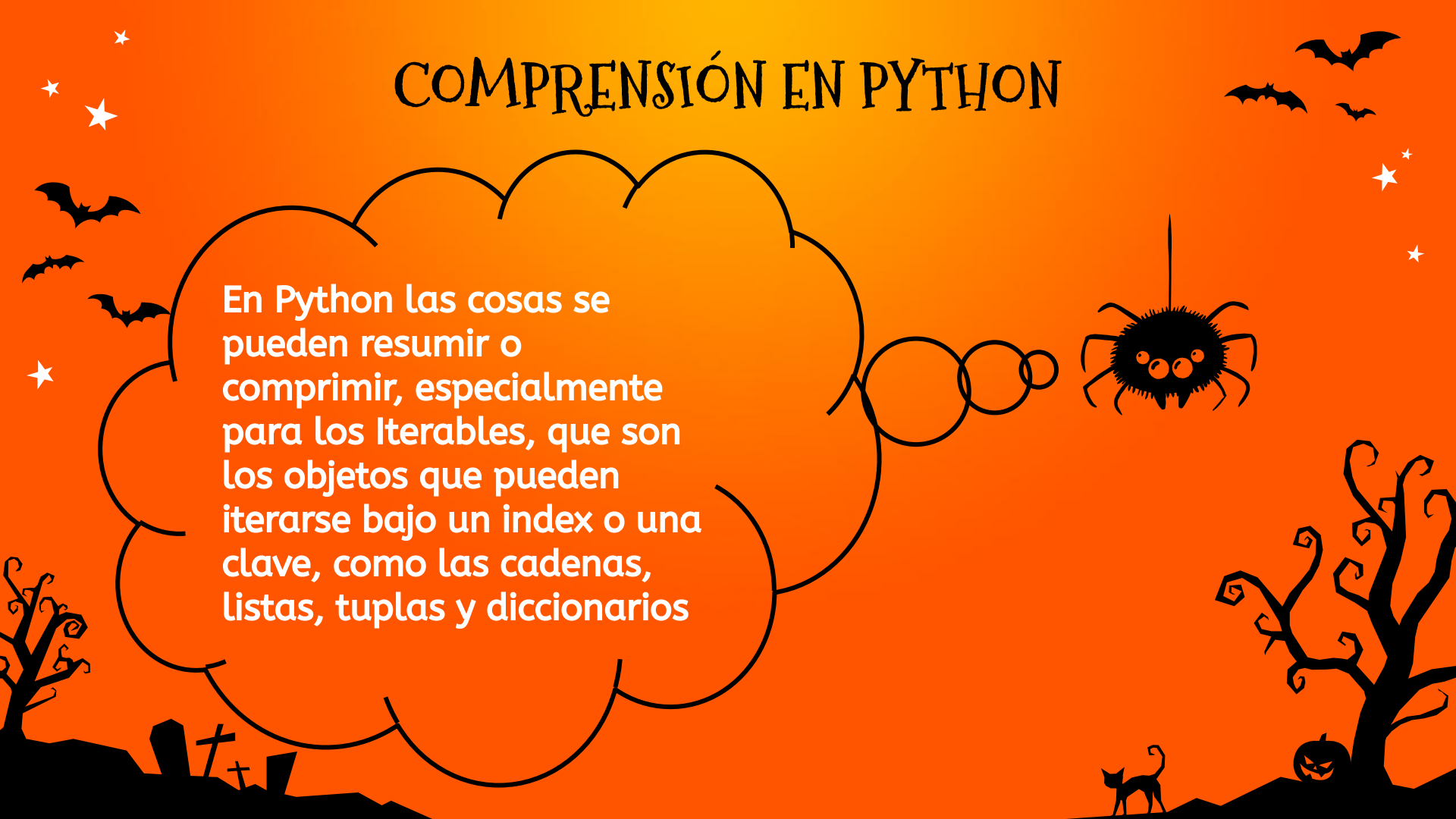
```
for letra in "Python":  
    if letra == "h":  
        break  
    print ("Letra actual : " + letra)
```

```
for letra in "Python":  
    if letra == "h":  
        continue  
    print ("Letra actual : " + letra)
```

```
for letra in "Python":  
    if letra == "h":  
        pass  
    print ("Letra actual : " + letra)
```

COMPRESIÓN EN PYTHON

En Python las cosas se pueden resumir o comprimir, especialmente para los Iterables, que son los objetos que pueden iterarse bajo un index o una clave, como las cadenas, listas, tuplas y diccionarios



COMPRESIÓN DE CICLOS

```
#Comprimir un if  
valor = 9  
if valor>0:  
    valor_es_positivo = True  
else  
    valor_es_negativo = False  
  
print(valor_es_positivo)  
  
valor_es_positivo = True if valor>0 else False  
print(valor_es_positivo)
```



COMPRESIÓN DE CICLOS

```
tupla = (1,2,3)
lista = [1,2,3]
cadena = "Hola Mundo"
dicc = {"key1":1, "key2":2}

print([valor for valor in tupla])
print([valor for valor in lista])
print([valor for valor in cadena])
print([(k,v) for valor in dicc.items()])
```

ITERABLES DE COMPREENSIÓN

```
lista = range(0,100)
pares = [valor for valor in lista if valor%2==0]
doble_pares_triple_impares [ 2*valor if valor%2==0 else
    3*valor for valor in lista]
```



Con estas
comprensiones puedo
hacer filtros, mapeos,
reducciones y demás
cosas con iterables



Built-in
Functions

Math
functions

¡SÓLO PUEDE USAR ESTAS FUNCIONES!
EN LA SIGUIENTE UNIDAD VEREMOS MÁS

MI CUARTO SCRIPT EN PYTHON

1. Escribir un script en python tal que dado un número natural n encuentre la suma de los enteros desde 1 hasta n .
2. Elabore un script en python que calcule el factorial de cualquier número entero positivo n .
3. Elabore un script en python que pase un número entero a binario.
4. Escribir un script en python para resolver el problema que dados los 3 coeficientes de un polinomio de segundo grado entregue los valores de las dos raíces (reales o imaginarias)
5. Escribir un script en python para que dado un número determine su conjunto numérico, si es par o impar y si es primo.

MI QUINTO SCRIPT EN PYTHON

Crear un script que le permita entrar al usuario un conjunto de números, siendo posible que existan valores repetidos y que me Regrese:

1. La media del conjunto de datos.
2. La desviación estándar.
3. La mediana del conjunto de datos.
4. La tabla de frecuencia del conjunto de datos.
5. La moda de del conjunto de datos.

MI SEXTO SCRIPT EN PYTHON

1. Un año es bisiesto si es múltiplo de 4, exceptuando los múltiplos de 100, que sólo son bisiestos cuando son múltiplos además de 400, por ejemplo, el año 1900 no fue bisiesto, pero el año 2000 si lo fue. Hacer un algoritmo que dado un año A nos diga si es o no bisiesto.
2. Realizar un algoritmo que dado la potencia de un binomio de newton sea capaz de determinar el coeficiente de cualquier componente del polinomio resultante.
3. Elaborar un algoritmo el cual reciba un número de N cifras y lo invierta.

MI SÉPTIMO SCRIPT EN PYTHON

1. Consideramos N puntos representados mediante (x, y) . Se desea realizar un algoritmo que permita obtener la recta de regresión que se ajuste a los puntos dados. Dicha recta vendrá dada mediante los valores b (interceptor), m (pendiente). Use la fórmulas de regresión.
2. Consideramos N puntos representados mediante (x, y) . Se desea realizar un algoritmo que permita obtener la parábola de regresión que se ajuste a los puntos dados. Dicha parábola vendrá dada mediante los valores a, b, c

MI OCTAVO SCRIPT EN PYTHON

Una encriptación de mensajes muy antigua inventada por Julio Cesar, la cual se basa en que a cada letra de un mensaje se le asocia un número (el orden en el abecedario), luego se suma al número de cada letra otro número (la clave) y el resultado es una nueva letra, mira en el abecedario. Por ejemplo, el mensaje “cruzad el rubicon”, se transformaría en “dsvabeUfmUsvcdpo” si le sumo el número clave 1 a cada letra (ya que la primera letra del mensaje cifrado, la ‘d’, es la que va tras la ‘c’, etc.). Note que la ch, ll, ñ no existen, y que después de la ‘z’ va la ‘a’. Además los espacios en blanco se convierten en letras U si la clave es 1, en V si es 2, etc. Para ello, elabore un algoritmo que a partir de un texto (cadena de caracteres leída por teclado) imprima texto cifrado con la clave entera, también leída desde teclado.

MI NOVENO SCRIPT EN PYTHON

El participante de un concurso tiene que recorrer en su automóvil una ruta determinada desde un lugar A a un lugar B, entre los cuales hay una distancia que es igual a D kilómetros. Con el depósito de gasolina lleno, su coche puede recorrer una distancia máxima de K metros. El concursante tiene un mapa de la ruta que debe recorrer en el que figuran las distancias entre las gasolineras que hay entre A y B, y planea realizar el viaje con la menor cantidad de paradas posible. Suponiendo que parte de A con el depósito lleno, y que la distancia entre las gasolineras obedecen la función $f(n)=0.05*D*(1-\exp(-n*0.1/D))$, siendo n es # de la gasolinera (distancia entre n-1 y n), desarrollar un algoritmo que determine en qué gasolineras deberá parar el concursante.

MI DÉCIMO SCRIPT EN PYTHON

1. Realice un script en python que usando la congruencia de Zeller permita ingresar una fecha y determina qué día de la semana fue.
2. Realice un script en python que encuentre las intersecciones entre tres rectas (si se da el caso) ingresando los valores de dos puntos por cada recta o ingresando las pendientes y los puntos de corte.
3. Dados los coeficientes de dos polinomios de grados N y M respectivamente, calcule la multiplicación de ambos polinomios e imprima los coeficientes del polinomio reducido resultante.



Happy Halloween!

¿Preguntas?

CREDITOS

Special thanks to all the people who made and released these awesome resources for free:



Presentation template by [SlidesCarnival](#)



Photographs by [Unsplash](#)