Python 3 (repaso)

Introducción a python 3 orientado a SOM

Python 3 (repaso)

- **ÍNDICE**
  - Conceptos básicos
  - Condicionales
  - While
  - For
  - Funciones
  - Librerías

Python 3 (repaso)

- ¿Por qué este curso de introducción?
- ¿Qué es python? enlace
- ¿Por qué hay diferentes versiones de python?¿Cuál usar?
- ¿Qué puedo hacer con python? enlace
- ¿Cuál es la curva de aprendizaje de python?
- ¿Tiene salida laboral programar con python? infojobs

- ¿Cuando empezamos?

¡YA!

Python 3 (repaso)

- ¿Por qué este curso de introducción?
  - Se busca dar una introducción básica a python para poder relacionarla con los scripts que se imparten en el CFGM SMR.
  - La programación de scripts que se imparte en el CFGM SMR son:
    - Shellscripts de GNU/Linux (bash).
    - Scripts de MSWindows (powershell).

Python 3 (repaso)

- ¿Cómo podemos ejecutar python?
  - Desde el terminal de python → ejecutando comandos

```
Python 3.11.1 (main, Dec  7 2022, 01:11:44) [GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hola')
Hola
>>>
```

  - Desde un script de python → extensión .py

    python nombre.py

```
sortiza@usuario-510-p108ns:~/python$ python3.11 01.py
¡Hola Mundo!
```

  - Si añadimos el shebang (#!/usr/bin/python3.11) daríamos permiso de ejecución al fichero nombre.py
  - ./nombre.py

```
sortiza@usuario-510-p108ns:~/python$ ./02.py
¡Hola Mundo!
```

Python 3 (repaso)

- Convención de nombres:
  - Clases empiezan con mayúsculas. El resto empiezan con minúsculas.
  - Si empieza con "_" es que es privado.
  - Si empieza con "__" es que es strong private.
  - Si acaba también con "__" es un nombre especial definido por el lenguaje.

Python 3 (repaso)

- Sangría
  - Siempre debemos aplicar una sangría para diferenciar bloques. Nosotros definimos la separación pero debemos mantener la misma sangría para los mismos bloques.

```
lugar = 'clase'

if lugar == 'clase':
    print('Logueando en el portal...')
else:
    print('No podemos acceder.')
print('¡Finalizado!')
```

Bien

```
lugar = 'clase'

if lugar == 'clase':
print('Logueando en el portal...')
else:
        print('No podemos acceder.')
print('¡Finalizado!')
```

Mal

Python 3 (repaso)

- Python cada vez que se encuentra un salto de línea considera que se trata de otro comando. Si un comando muy largo queremos escribirlo en dos líneas podemos usar “\”.

```
Comando \
Sigue comando
```

- Si el comando tiene [], {}, o () no es necesario usar “\” para juntar dos líneas.

Python 3 (repaso)

- Python soporta ' ', " ", ''' ''' y """ """. La triple comilla, simple o doble se usa para expandir cadenas por varias líneas.

```
cadena="""Probando a escribir una cadena
multilína"""
```

- **COMENTARIOS**
  - Al igual que en bash, los comentarios se hacen con #. Python no tiene comentarios multilínea, hay que hacer cada línea.

```
# Script número 04
```

- Usando el ";", podemos hacer posible ejecutar varios comandos en una misma línea.

```
print("Hola");print("Adios")
```

Python 3 (repaso)

ej01.py

- Nuestro primer script de python. Mostrar información por pantalla.
- Método uno de ejecución
  - chmod 755 ej01.py
  - ./ej01.py
- Método dos de ejecución
  - python3.11 ej01.py

```
#!/usr/bin/python3.11
print("Hola mundo")
```

```
thub/cursoBa
Hola mundo
```

NOTA: Desde python 3.6
print(f“Tienes {edad} años.”)

Python 3 (repaso)

ej02.py

- Recoger información por teclado

```
#!/usr/bin/python3.11
print("Indique su nombre: ")
nom = input()
print(nom)
cog = input("Indique su primer apellido: ")
print(cog)
print()
```

```
Indique su nombre:
Alberto
Alberto
Indique su primer apellido: Pérez
Pérez
```

Python 3 (repaso)

ej03.py

- Repasemos de nuevo los comentarios.

```
#!/usr/bin/python3.11

print("Comentarios en el código")

# Commentario de una línea
```

Python 3 (repaso)

ej04.py
ej04b.py

- Secuencia de comandos
  - Ejemplo 1

```
#!/usr/bin/python3.11
print("Primer") ; print("Segundo")
```

  - Ejemplo 2

```
#!/usr/bin/python3.11

import sys; x='foo'; sys.stdout.write(x + '\n')
```

Python 3 (repaso)

ej05.py

- **Variables**
  - En python tenemos diferentes tipos de variables. Sólo vamos a usar los indicados en la siguiente imagen durante el curso. Para mas información, el curso intermedio de python.

```
contador = 100 # Entero
miles = 1000.0 # Float
nombre = "Alberto" # Cadena
```

  - Podemos extraer ante la duda el tipo de variable.

```
print(type(nombreVariable))
```

Python 3 (repaso)

ej06.py

- **Variables**
  - Podemos borrar variables si ya no queremos que sean accesibles o liberar espacio.

```
# Números. Python soporta enteros, float y complejos.
edad = 18
peso = 77.6

print (edad)
# Se pueden borrar variables usando el comando "del"
del edad
print (edad)
```

Python 3 (repaso)

- **Operaciones matemáticas**

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

Python 3 (repaso)

Act01.py

- Realiza un script de python en el cual se pida dos números y se muestre por pantalla la suma de los números insertados.

Python 3 (repaso)

Act01.py

- Solución

```
#!/usr/bin/python3.11
num1 = int(input("Indique el primer número: "))
num2 = int(input("Indique el segundo número: "))
resultado = num1 + num2
print("La suma de ",num1," y ",num2," es igual a ",resultado)
```

  - Error común

```
#!/usr/bin/python3.11
num1 = input("Indique el primer número: ")
num2 = input("Indique el segundo número: ")
resultado = num1 + num2
print("La suma de ",num1," y ",num2," es igual a ",resultado)
```

Python 3 (repaso)

Act01b.py

- Realiza de nuevo la Act01.py pero se debe mostrar además de la suma, la resta, multiplicación y división.

Python 3 (repaso)

Act02.py

- Realiza un formulario que muestre por pantalla los datos de manera ordenada que se pregunten a continuación:
  - Nombre
  - Apellidos
  - Edad
  - Dirección y
  - Peso

Python 3 (repaso)

ej07.py

- **Cadenas**
  - Cuando tenemos una cadena podemos tratar la información que ella contiene de diferentes maneras. Mostramos algunas. ¿Puedes explicarlas?

```
#!/usr/bin/python3.11

cad = "Hola mundo"
edad = 18
nota = 10
print(cad)
print(cad[0])
print(cad[2:5])
print(cad*2)
print(cad + ", añado") # Concatenamos cadenas
print(cad , ", añado II") # También nos concatena.
```

Python 3 (repaso)

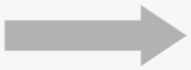
ej08.py

- **Cadenas**
  - Concatenar etiquetas usando “+”
  - Multiplicar una cadena “*”

```
#!/usr/bin/python3.11

cadena1 = "Hola "
cadena2 = "Mundo"
cadena3 = cadena1 + cadena2
print(cadena3)


print(cadena1*3)
```



```
thub/cursoBasicol
Hola Mundo
Hola Hola Hola
```

Python 3 (repaso)

Act03.py

- Disponemos de un dato capturado por nuestro sistema en una variable de tipo cadena. El dato se muestra a continuación:

  dato=”FA:BA:DA:FA:BA:DA 192.168.5.3”

  - Queremos extraer la IP y mostrarla por pantalla.

Python 3 (repaso)

Act03.py

- **Solución**

```
#!/usr/bin/python3.11
# Disponemos de un dato capturado en una cadena.
datos="FA:BA:DA:FA:BA:DA 192.168.5.3"
print(datos)
ip=datos[18:]
print(ip)
print(datos[18:])
```

Python 3 (repaso)

Act04.py

- A partir de la actividad Act02.py, ampliará haciendo un ping a dicha máquina para comprobar si está encendida o no (suponemos que no hay restricciones a ping).

Python 3 (repaso)

Act04.py

- **Solución**
  - Aquí hay varias cosas mejorables. Fijémonos a grandes rasgos.

```
# Importamos librerías
import os

datos="FA:BA:DA:FA:BA:DA 192.168.5.192"
ip=datos[18:]

# Ejecutamos sobre el terminal del sistema
response = os.system("ping -c 5 " + ip + " >/dev/null")
if response == 0:
    print("Ordenador encendido")
else:
    print("Ordenador apagado")
print("Saliendo...")
```

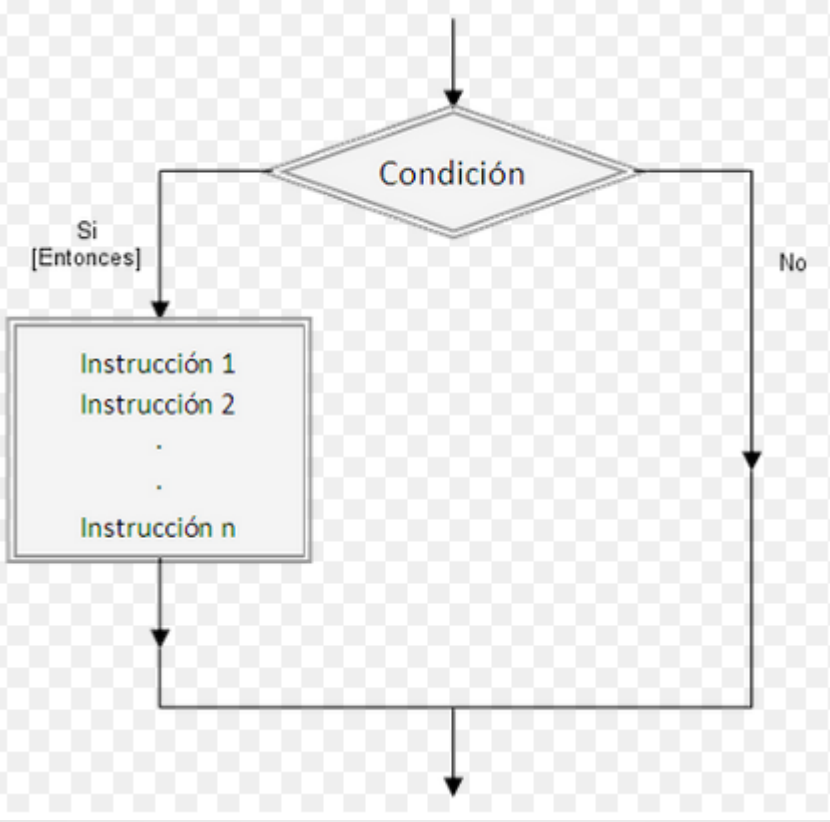
Python 3 (repaso)

- **Comparaciones**

Operador	Descripción	Ejemplo
==	¿son iguales a y b?	`>>> 5 == 3` `False`
!=	¿son distintos a y b?	`>>> 5 != 3` `True`
<	¿es a menor que b?	`>>> 5 < 3` `False`
>	¿es a mayor que b?	`>>> 5 > 3` `True`
<=	¿es a menor o igual que b?	`>>> 5 <= 5` `True`
>=	¿es a mayor o igual que b?	`>>> 5 >= 3` `True`

Python 3 (repaso)

- **Condicionales**
  - Los condicionales parten de permitir controlar el flujo de ejecución dependiendo de si se cumple o no una condición.



Python 3 (repaso)

- **Condicionales**

```
if [boolean expression]:
    statement1
    statement2
    ...
    statementN
```

Python 3 (repaso)

ej09.py

- **Condicionales**

```
#!/usr/bin/python3.11

edad = int(input("Dime tu edad: "))

if edad >= 18:
    print("Eres mayor de edad")
```

Además tenemos las palabras **and** y **or**

```
if a > b and b < c:
    print("Se cumplen las dos condiciones")
```

```
if a > b or b < c:
    print("Si se cumple una de las dos condiciones")
```

Python 3 (repaso)

Act05.py

- Realiza un script que te pregunte tu edad. Tanto si eres mayor de edad como si no, debe mostrarte un mensaje indicándote que puedes sacarte el carné de conducir o no.

Python 3 (repaso)

- **Condicionales**

```
if [boolean expression]:
    statement1
    statement2
    ...
    statementN
else:
    statement1
    statement2
    ...
    statementN
```

Python 3 (repaso)

ej10.py

- **Condicionales**

```
#!/usr/bin/python3.11

edad = int(input("Indica tu edad: "))

if edad < 18:
    print("Eres menor de edad")
else:
    print("Eres mayor de eddad")
```

Python 3 (repaso)

Act05b.py

- Realiza un script que te pregunte tu edad. Tanto si eres mayor de edad como si no, debe mostrarte un mensaje indicando si se te permite sacar el carné de conducir o no.

Python 3 (repaso)

- **Condicionales**

```
if [boolean expression]:
    [statements]
elif [boolean expresion]:
    [statements]
elif [boolean expresion]:
    [statements]
else:
    [statements]
```

Python 3 (repaso)

ej11.py

- **Condicionales**

```
#!/usr/bin/python3.11

nom = input("Indica tu nombre: ")

if nom == "Moto":
    print("Suzuki")
elif nom == "Marys":
    print("Gold")
else:
    print("No se sabe")
```

Python 3 (repaso)

Act06.py
Act06b.py

- Realiza un script que te pregunte tu edad. Tanto si eres mayor de edad como si no, debe mostrarte un mensaje indicando si se te permite sacar el carné de conducir o no.
- Si insertas un valor que no sea numérico debe indicar un mensaje de error para avisar al usuario.

Python 3 (repaso)

Act07.py
Act08.py
Act09.py

- **ACTIVIDADES**
  - Deseamos generar un script que nos permita el acceso al sistema. El script no solicitará una contraseña y si es correcta, nos mostrará un mensaje de bienvenida. Si no lo es, mostrará un aviso de alerta.
  - Deseamos generar un script que nos solicite dos números y luego nos pregunte si queremos que se haga una suma, resta o multiplicación. Mostrará por pantalla el resultado de lo que hayamos seleccionado.
  - Deseamos generar un script que dado un número de DNI (sin letra), nos indique cuál es la letra asociada a ese número.

Python 3 (repaso)

- ¿Cómo calcular la letra del DNI?

  https://www.adslzone.net/como-se-hace/internet/letra-dni-cif-nif/

Python 3 (repaso)

Act10.py
Act11.py
Act12.py

- **ACTIVIDADES.** Usa el condicional adecuado en cada caso.
  - Un programa que te pida un número y te diga si es par o impar.
  - Para tributar un determinado impuesto se debe ser mayor de 16 años y tener unos ingresos iguales o superiores a 1000 € mensuales. Escribir un programa que pregunte al usuario su edad y sus ingresos mensuales y muestre por pantalla si el usuario tiene que tributar o no (Fuente: Internet).
  - Escribir un programa para una empresa que tiene salas de juegos para todas las edades y quiere calcular de forma automática el precio que debe cobrar a sus clientes por entrar. El programa debe preguntar al usuario la edad del cliente y mostrar el precio de la entrada. Si el cliente es menor de 4 años puede entrar gratis, si tiene entre 4 y 18 años debe pagar 5€ y si es mayor de 18 años, 10€ (Fuente: Internet)
  -

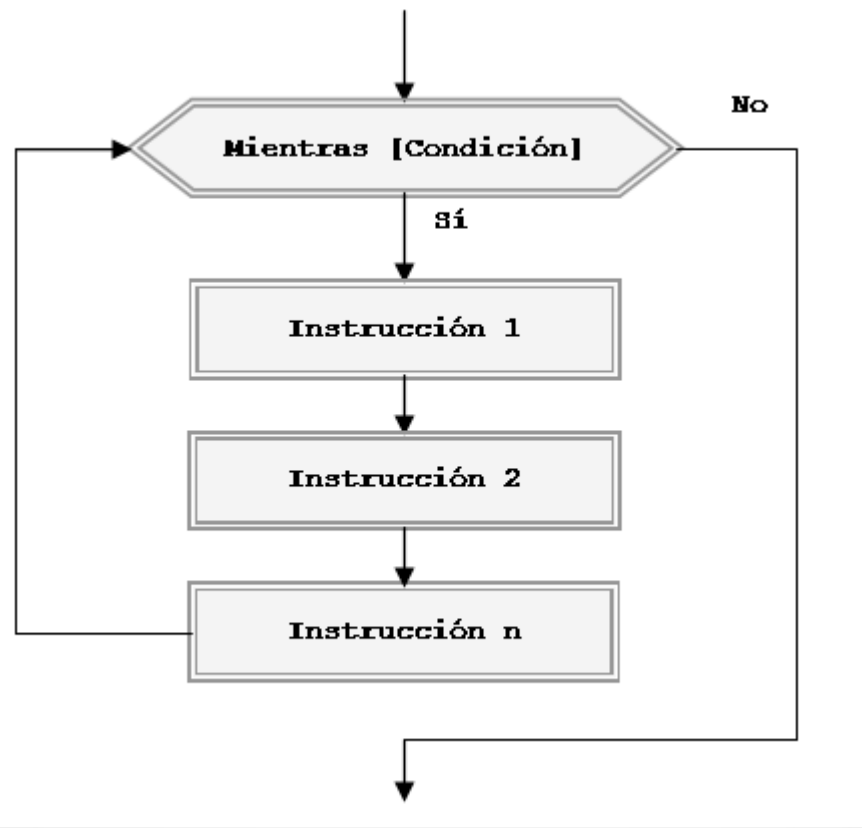
Python3 (repaso)

Act13.py

- **ACTIVIDADES.** Usa el condicional adecuado en cada caso.
  - Dado un número, cuente el número total de dígitos de un número. Ej 1232, cuatro dígitos.

Python 3 (repaso)

- **While**



Python 3 (repaso)

ej12.py
ej13.py

- **While**
  - ¿Qué mostraría por pantalla estos dos scripts?

```
i = 1
while i < 6:
    print (i)
    i = i + 1
```

```
i = 1
while i < 6:
    print (i)
```

Python 3 (repaso)

ej14.py

- **While**

```
#!/usr/bin/python3.11

i = 1
while i < 6:
    print(i)
    i = i + 1
else:
    print("i ya vale mas de lo deseado")
```

Python 3 (repaso)

ej15.py

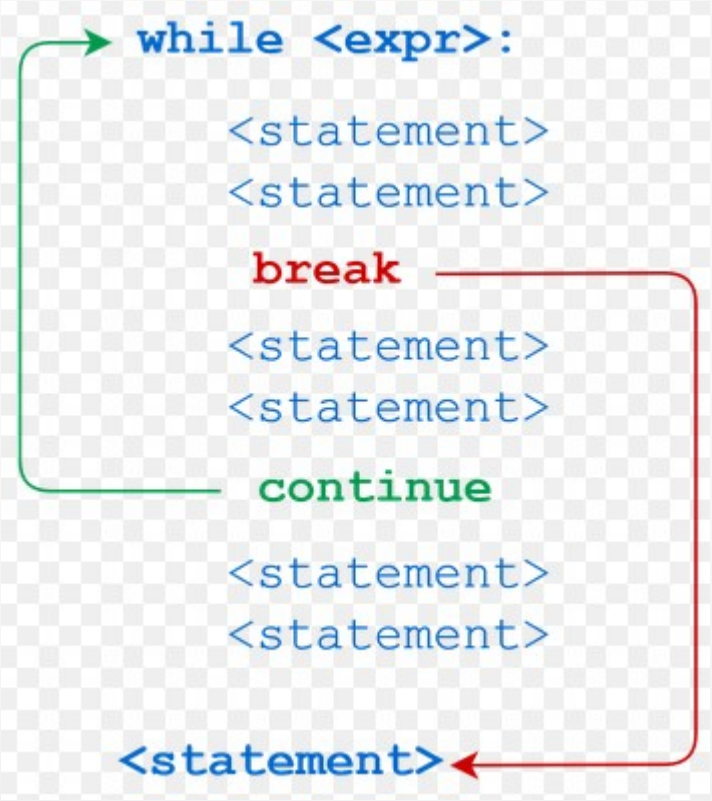
- **While**

```
#!/usr/bin/python3.11

i = 1
while i < 6:
    print(i)
    if i == 3:
        break # Saldrá del bucle donde nos encontramos
    i = i + 1
```

Python 3 (repaso)

- **continue y break (también tenemos pass)**



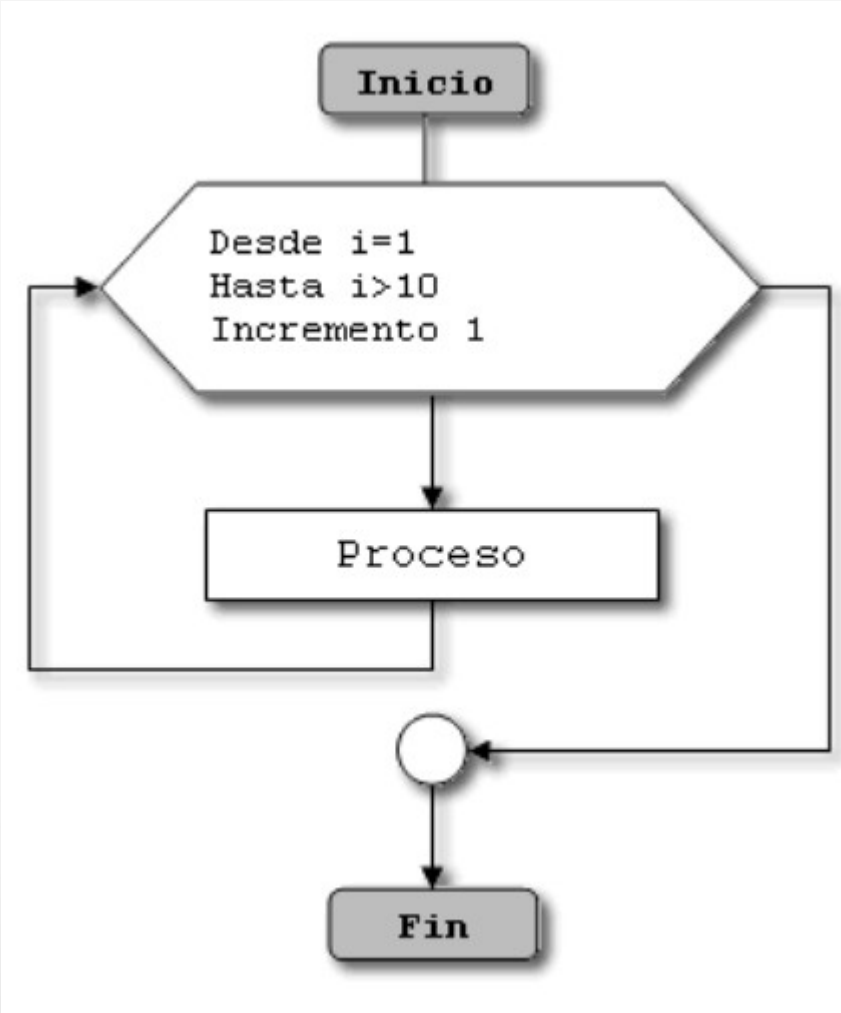
Python 3 (repaso)

Act14.py
Act15.py
Act16.py

- **ACTIVIDADES**
  - Realiza un script en el cual le indique un número del uno al diez y te muestre desde ese número hasta el cero, de uno en uno. Ej el 5, mostraría 5, 4, 3, 2, 1 y 0.
  - Realiza un script que solicite un número del uno al diez y muestre su tabla de multiplicar.
  - Realiza un script que muestre por pantalla el mensaje “Bienvenido a python” y luego el mensaje “Nos vemos…”, repitiéndose de manera ininterrumpida.

Python 3 (repaso)

- **For**



Python 3 (repaso)

ej16.py
ej17.py

- **For**

range(valor inicial, valor final, step)

range(número)

```
for i in range(5):
    print(i)
```

```
for i in range(10, 18, 3):
    print(i)
```

¿Qué diferencia a la hora de ejecutar estos dos códigos nos encontramos?

Python 3 (repaso)

ej18.py

- **For**
  - Un nuevo tipo de variable. ¿Cómo se llama?
  - ¿Utilidades? Busca en la información oficial

```
frutas = ['Manzana', 'Pera', 'Plátano']
for i in frutas:
    print (i)
```

Python 3 (repaso)

Act17.py
Act19.py
Act20.py

- **ACTIVIDADES**
  - Escribe un script al cual le indicarás cinco números y cada vez que introduzcas uno debe decirte si es mayor, menor o igual a diez.
  - Crear un bucle que cuente todos los números pares hasta el 100 y los muestre por pantalla.
  - Imprima el siguiente patrón con el ciclo for:

```
5 4 3 2 1
4 3 2 1
3 2 1
2 1
1
```

Python 3 (repaso)

ej19.py

- **Funciones**
  - Las funciones son muy útiles cuando necesitamos ejecutar en varias ocasiones el mismo trozo de código. Se definen con la palabra reservada “def”.
  - Sólo vamos a ver algunas implementaciones. Para el resto, véase el curso intermedio.

```
def nombreFuncion():
    código

nombreFuncion()
```

```
def mensaje():
    print("Hola")

mensaje()
```

Python 3 (repaso)

ej20.py

- **Funciones**
  - Cuando definimos una función podemos indicar que debe recibir datos para hacer algo. Véase un ejemplo a continuación.

```
#!/usr/bin/python3.11

def suma(num1, num2):
    resultado = num1 + num2
    print(resultado)

n1 = int(input("Número 1: "))
n2 = int(input("Número 2: "))
suma(n1, n2)
```

  - Fijaros tanto en la definición de la función, como la llamada y el uso de int().

Python 3 (repaso)

ej21.py
ej22.py

- **Built-in functions**
  - Además de las funciones que podamos crear, existen ya definidas por python muchas otras. Véamos algunas a modo de ejemplo. Len y pow

```
#!/usr/bin/python3.11

cad = "Prueba"
x = len(cad)
print(x)
```

```
#!/usr/bin/python3.11

n1 = 3
y = pow(n1, 3)
print(y)
```

Python 3 (repaso)

- **Librerías**
  - Una ventaja de python es que existen multitud de librerías con código que podemos reutilizar. Nosotros podemos crear nuestras propias librerías y compartirlas o usar las de otros programadores.
  - En nuestro caso vamos a utilizar alguna que puede venirnos muy bien al trabajar con sistemas operativos.

Python 3 (repaso)

ej23.py

- **Librería colorama**
  - Instalamos la librería con **pip install colorama**

```
#!/usr/bin/python3.11

from colorama import init, Fore, Back, Style

init()
print(Fore.RED + "Texto en color rojo")
print(Back.WHITE + "Texto rojo con fondo blanco")
print(Back.WHITE + Style.BRIGHT + "Texto brilla")
print(Style.RESET_ALL + "Texto con valores por defecto")
print(Fore.WHITE+Back.BLUE + "Texto blanco con fondo azul" + Back.RESET)
print("Texto blanco sobre fondo negro")
```

```
Texto color rojo
Texto color rojo sobre fondo blanco
Txt rojo brill.s/blanco
Texto con valores por defecto
Texto blanco sobre azul
Texto blanco sobre fondo negro
```

Python 3 (repaso)

ej24.py

- **Librería platform**

```
#!/usr/bin/python3.11

import platform

sistema = platform.system()
print("Estamos en " + sistema)
```

Python 3 (repaso)

ej25.py
ej25b.py
ej25c.py

- **Librería subprocess** enlace

```
#!/usr/bin/python3.11

from subprocess import call
call('clear')
```

```
#!/usr/bin/python3.11

from subprocess import call
comando_y_argumento = ['ls', '-lha']
call(comando_y_argumento)
```



```
#!/usr/bin/python3.11

import shlex, subprocess

command_line = 'sudo apt-get install sl -y'

args = shlex.split(command_line)

subprocess.call(args)
```

Python 3 (repaso)

Act21.py
Act22.py
Act23.py

- **ACTIVIDADES**
  - Realiza un script que cree un usuario llamado alumno.
  - Realiza un script que cree un usuario llamado alumno2 independientemente del SO que estemos usando.
  - Realiza un script que ejecute el comando ls y muestre por pantalla un mensaje personalizado indicando si se ejecutó bien o mal el comando.

Python 3 (repaso)

Fin