

## Lenguaje Python Versión 3 (3.x.y)

- Python oficial de python.org
- Anaconda Python
- Thonny
- WinPython
- Python(x,y)
- Pydroid 3 (Android)

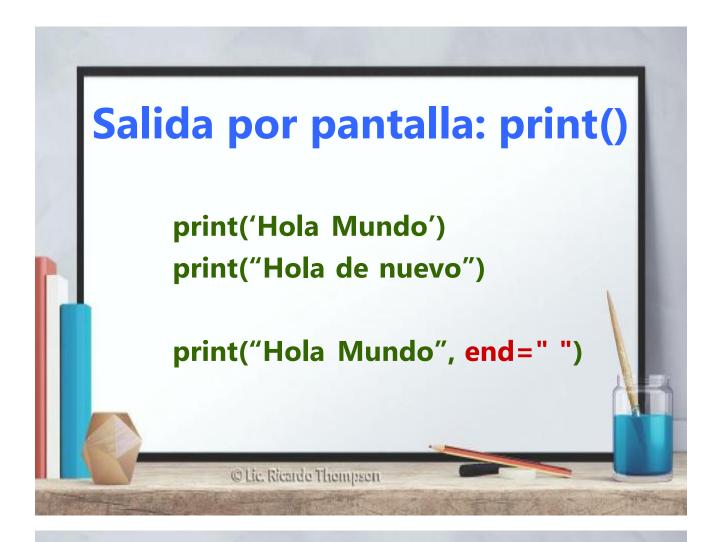
## ¿Qué es Python?

- Es un lenguaje de muy alto nivel.
- Es interpretado: No se compila.
- Es multiparadigma.
- El espíritu de Python ("zen") privilegia la legibilidad del código.
- Es portable.
- Es sensible a mayúsculas y minúsculas.

@ Lie. Ricarde Thempsen

# Estructura de los programas en Python

- No tienen un formato rígido.
- No hay llaves ni puntos y coma.
- Las instrucciones terminan con Enter.
- Los comentarios comienzan con el signo numeral: #



- No se declaran.
- Se crean automáticamente con la asignación de un valor inicial.
- Intentar usar una variable no inicializada provoca un error.

Reglas para crear nombres de variables:

- Sólo letras, números y el guión bajo.
- No pueden comenzar con un número.
- No pueden coincidir con las palabras reservadas del lenguaje.

@ Lic. Ricarde Thempsen

### **Variables**

- Los nombres de variables <u>deben</u> tener sentido.
- Deben evitarse variables llamadas "l" (ele minúscula) u "o" (letra o) porque pueden confundirse fácilmente con números.

- Pueden almacenarse números (enteros y reales) y textos.
- Existen valores booleanos: True y False.
- Se pueden hacer asignaciones múltiples
  - $\sqrt{a=b=c=0}$
  - $\checkmark$  a, b, c = 3, "Lunes", 5.18

@ Lie. Ricarde Thempsen

## **Variables**

 Para imprimir variables se las separa de las constantes con una coma:

 Python agrega automáticamente un espacio como separador.

 Puede lograrse mayor control de la salida impresa con el operador %:

```
cant = 10
precio = 12.5
print("Cantidad: %6d" %cant) # Cantidad: 10
print("Precio: %5.2f" %precio) # Precio: 12.50
```

No va coma luego de cerrar comillas.

@ Lic. Ricarde Thempsen

## **Variables**

- Los especificadores de conversión disponibles son:
  - %d para números enteros
  - %f para números reales
- Los números ubicados luego del signo de porcentaje afectan a la salida impresa.
  - El primero es el ancho total del número.
     El segundo, la cantidad de decimales.

 Puede escribirse un 0 delante del ancho para rellenar con ceros.

print("%04d" %a) → 0003

 Si hay más de una variable, éstas deben encerrarse entre paréntesis:

print("X = %4d - Y = %4d" %(x, y))

@ Lie. Ricarde Thempsen

### **Variables**

 Para ingresar valores por teclado se utiliza la función input():

a = input("Mensaje")

input() <u>siempre</u> devuelve un string.



Existen funciones para convertir este string a otros tipos de dato:

```
n = int(input("Mensaje"))
```

r = float(input("Mensaje"))

@ Lie. Ricarde Thempsen

## **Operadores aritméticos**

+ Suma

- Resta
- \* Multiplicación / División real
- // División entera % Módulo o resto
- \*\* Potenciación

#### Orden de evaluación

- 1. Potenciación
- 2. Menos unario
- 3. Multiplicación, división y módulo
- 4. Suma y resta

Qué resultado arroja -2\*\*2?

© Lie. Ricarde Thempsen

## Asignación extendida

$$a += 1 \Leftrightarrow a = a + 1$$

$$a = 2 \Leftrightarrow a = a - 2$$

$$a *= 3 \Leftrightarrow a = a * 3$$

$$a/=4 \Leftrightarrow a=a/4$$

$$a //= 5 \Leftrightarrow a = a // 5$$

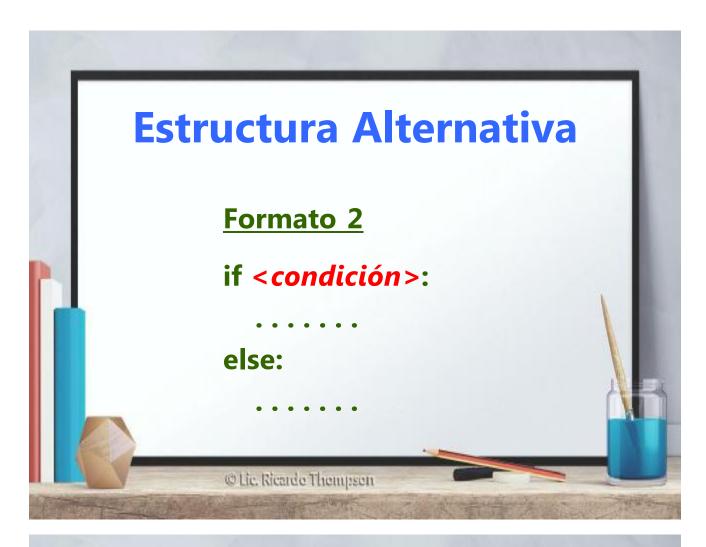
$$a \% = 6 \Leftrightarrow a = a \% 6$$

$$a^{**}=7 \Leftrightarrow a=a^{**}7$$

No existen operadores incrementales.









#### **Estructura Alternativa**

- Las condiciones y el else van seguidas del carácter "dos puntos".
- La sangría o indentación es lo que establece el alcance del if.
- Python recomienda una sangría standard de 4 espacios, sin tabs.
  - La sangría debe ser uniforme.

@ Lic. Ricarde Thempsen

## **Operadores relacionales**

== igual

> mayor

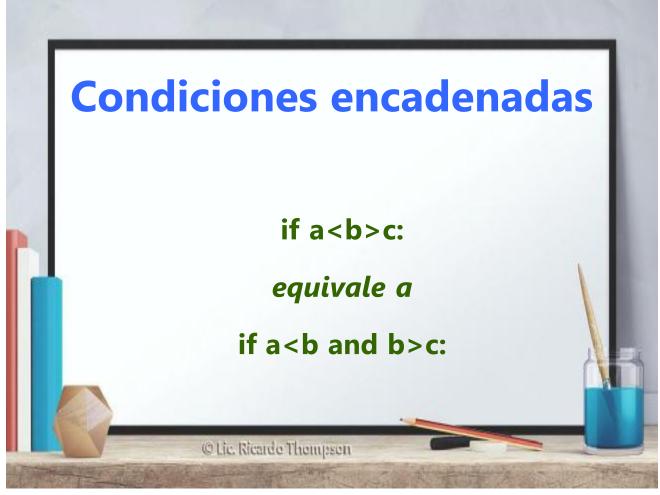
< menor

>= mayor o igual

<= menor o igual

!= distinto







Cuidado con las condiciones encadenadas:

if 
$$n > = 10 < = 100$$
:

equivale a

if n >= 10 and 10 <= 100:

La variable debe escribirse en el medio:

@ Lic. Ricarde Thempsen

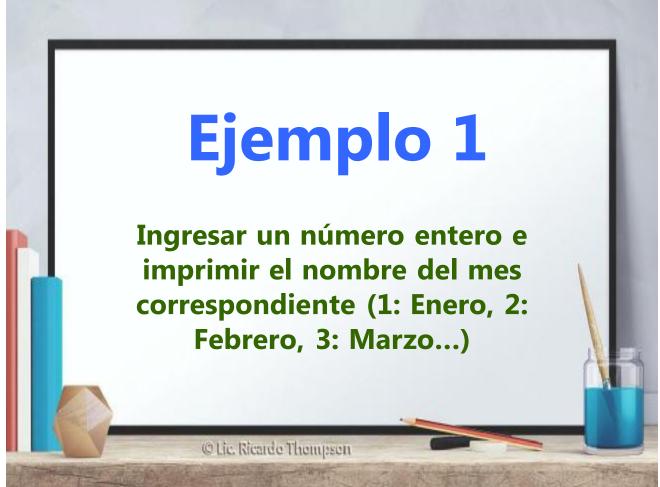
## **Operador condicional**

<var>= <valor1> if <condición> else <valor2>

**Ejemplo:** 

a = b if b > = 0 else -b





```
mes = int(input("Mes?"))
if mes == 1:
    print("Enero")
elif mes == 2:
    print("Febrero")
elif mes == 3:
    print("Marzo")
[....]
elif mes == 12:
    print("Diciembre")
else:
    print("Mes inválido")
```



#### **Estructura Iterativa**

- La condición va seguida del carácter "dos puntos".
- La sangría o indentación es lo que establece el alcance del ciclo.
- Python recomienda una sangría standard de 4 espacios, sin tabs.
  - La sangría debe ser uniforme.

@ Lic. Ricardo Thempsen

#### **Estructura Iterativa**

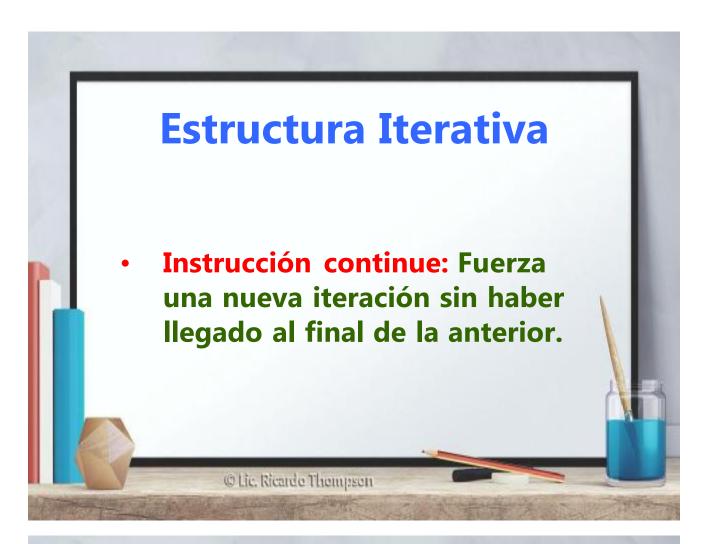
- Instrucción break: Abandona el último ciclo ejecutado.
- Cláusula else: El código de esta cláusula sólo se ejecuta si el ciclo terminó en forma normal, es decir sin break.

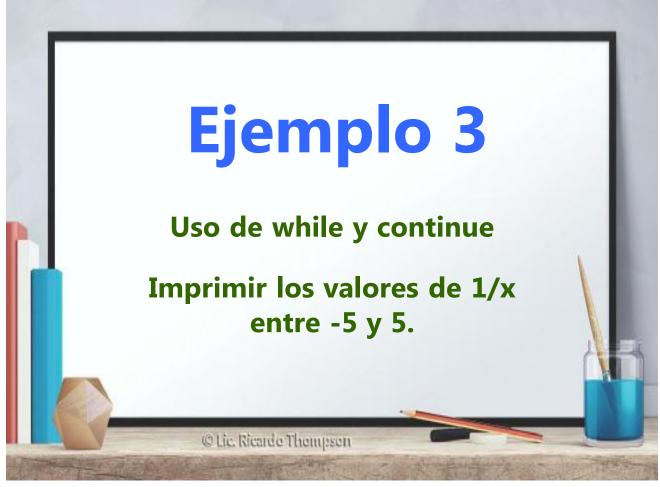
## Ejemplo 2

Uso de while, break y else:

Leer un número entero e imprimir un mensaje indicando si se trata de un número primo o no.

```
n = int(input("Ingrese un número: "))
divisor = 2
while divisor<n:
    if n % divisor == 0:
        print(n, "no es un número primo")
        break
    divisor = divisor + 1
else:
    print(n, "es un número primo")
```





```
x = -6
while x<5:
    x = x + 1
    if x == 0:
    continue
    print("%2d: %5.2f" %(x,1/x))
```



#### **Estructura Iterativa**

- for se utiliza para recorrer una secuencia.
- Esta secuencia puede ser cualquier *iterable*: Un rango, una lista, una cadena, un archivo, una tupla, un conjunto o un diccionario.
- Las instrucciones break y continue hacen lo mismo que en while.

@ Lic. Ricarde Thempsen

## Función range()

- Genera una secuencia de números enteros.
- Admite tres formas de utilización.

## Función range()

#### Formato 1:

#### range(<vfinal>)

- Genera una secuencia de números enteros entre 0 y <vfinal>.
- <vfinal> no está incluido.

@ Lic. Ricarde Thempsen

### Función range()

#### Formato 2:

range(<vinicial>, <vfinal>)

- Genera una secuencia de enteros entre <vinicial> y <vfinal>.
- <vfinal> no está incluido.

## Función range()

#### Formato 3:

range(<vinicial>, <vfinal>, <inc>)

- Genera una secuencia de enteros entre <vinicial> y <vfinal> con incremento <inc>.
  - <vfinal> no está incluido.

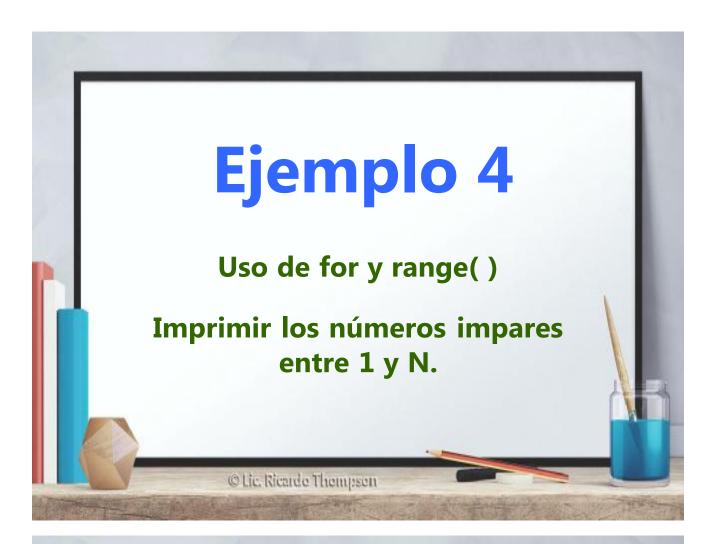
@ Lie. Ricarde Thempsen

### Función range()

#### Formato 3:

range(<vinicial>, <vfinal>, <inc>)

El incremento puede ser negativo



```
# Imprimir los números impares entre 1 y N

n = int(input("Ingrese un número: "))
for i in range(1, n+1, 2):
    print(i, end=" ")
print()
```

## **Ejercitación**

- Buscar en Internet el Zen de Python y analizarlo.
- Escribir un programa para resolver el antiguo acertijo chino que se describe a continuación:

"He contado 35 cabezas y 94 patas entre las gallinas y los conejos de mi granja. Cuántos conejos y cuántas gallinas tengo?"

(Sugerencia: Verificar todas las combinaciones posibles hasta hallar la correcta; no realizar una resolución algebraica).