# Clase 1 - Parte 1: Programación

Introducción a la Programación.

# Temario

- ¿Qué es Python?
  Lenguajes de programación.  Empezando a programar en Python:
  - Variable
  - Tipos de Datos
  - Estructuras de control:
    - Condicionales:
      - if
      - if-else
      - if-elif-else.

# ¿Qué es Python

"Python es un lenguaje de programación que permite **escribir código rápido** e **integrar sistemas** de forma eficiente".

# LENGUAJES DE PROGRAMACIÓN

Entonces, ¿qué es un lenguaje de programación?

- Un **lenguaje de programación** es un **lenguaje formal** que se utiliza para desarrollar **programas de software**.
- Por ejemplo: Aplicaciones, páginas webs, scripts u otros conjuntos de instrucciones.
- Los programas pueden ser ejecutados por una computadora, **permitiendo controlar su comportamiento, el de dispositivos periféricos** y en algunos casos proveyendo la interfaz de usuario.

# LENGUAJES DE PROGRAMACIÓN

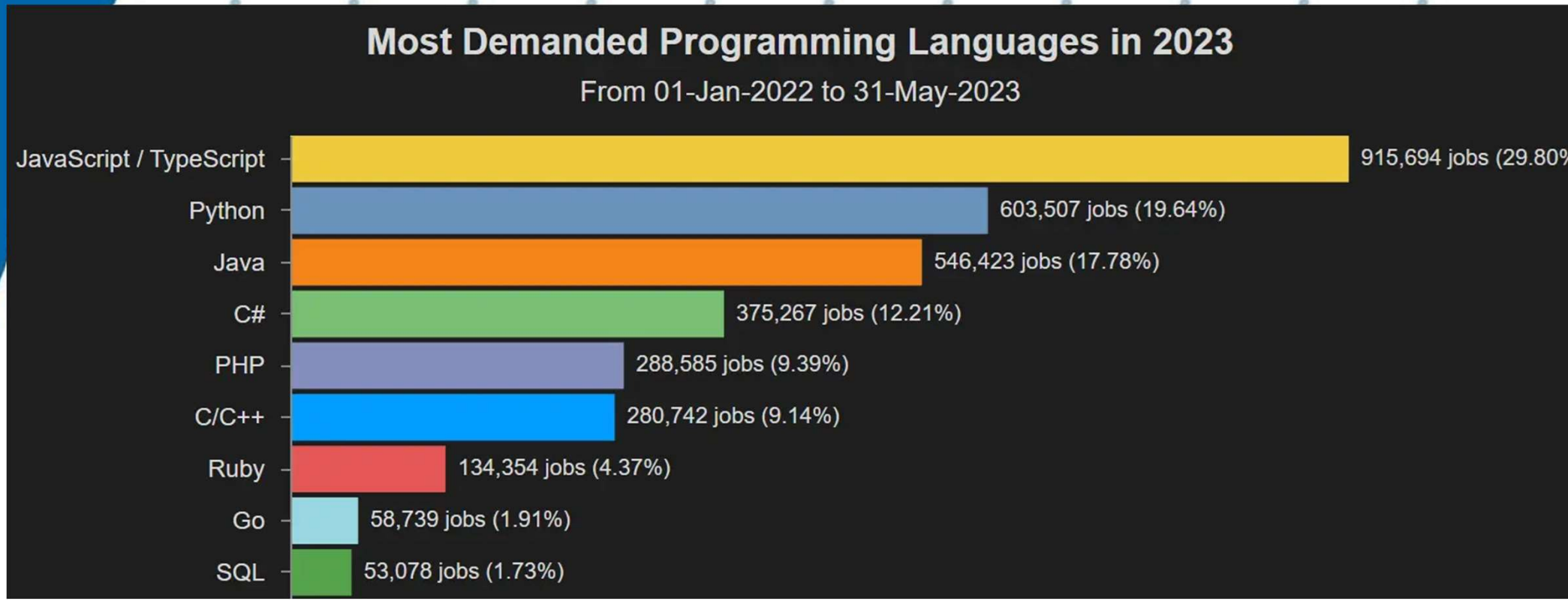
Existe una **amplia variedad de lenguajes** de programación.

Cada uno de ellos posee un conjunto de **reglas sintácticas** y **semánticas** asociadas que determinan la forma de expresar las instrucciones de programa.

Algunos **lenguajes de programación** son **multipropósito**: se pueden utilizar para desarrollar aplicaciones de escritorio, desarrollos web, aplicaciones móviles, etc…

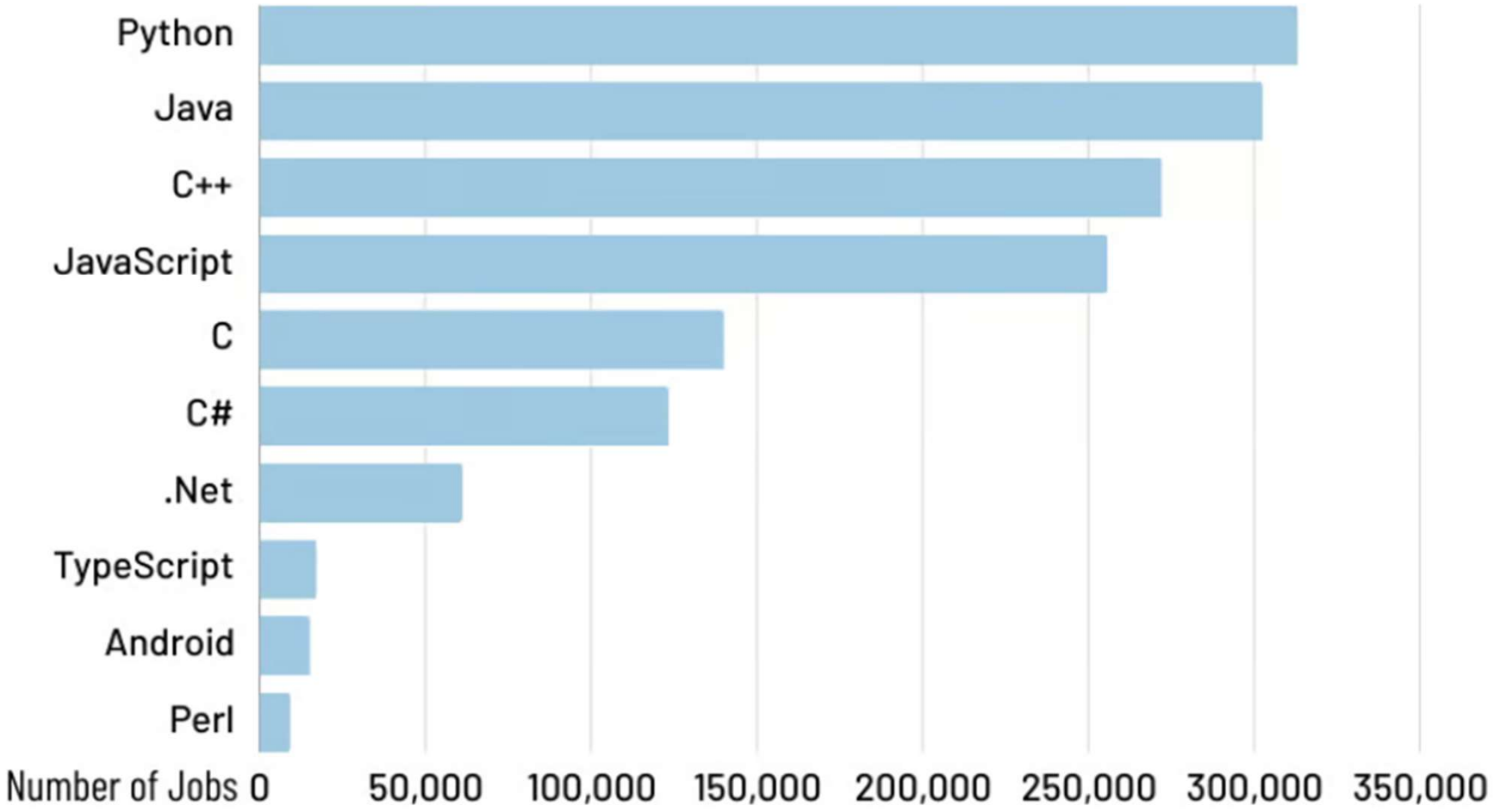
A la hora de **elegir un lenguaje** para programar **dependerá de la aplicación** que se va a desarrollar y **del soporte** que este provee.

# Ranking 2023
# según: devjobsscanner



# Most in-demand programming languages of 2024

Based on LinkedIn job postings in the US




By: CodingNomads

# ¿Por qué elegir Python?

Está desarrollado bajo una licencia de código abierto, por lo que es de libre uso y distribución.

Es uno de los lenguajes de programación **más versátiles** y puede ser usado en **muchos campos diferentes**: permite programar desde **videojuegos** hasta **aplicaciones móviles**.

# Por qué elegir Python?

- **Lenguaje interpretado** (no requiere compilación para ser ejecutado)**.**
- Con **tipado dinámico** (cambio de tipo en las variables durante ejecución).
- **Multiplataforma** (Linux, Dos, Windows, etc).
- Es **multiparadigma**: **O.O.** pero admite programación **imperativa y funcional**.
- Con **gran cantidad de librerías** predefinidas.
- Diseñado para ser **rápido de aprender**, de usar, comprender y posee una **sintaxis simple**.

# Antes de

Bajar Python de la página oficial:

https://www.python.org/downloads/

Alternativas:

Google Colaboratory → https://colab.research.google.com/

Programiz → https://www.programiz.com/python-programming/online-compiler/

# Comenzando a programar en Python

- Una **variable** es un nombre que representa o refiere a espacio de memoria donde se almacena un dato.
- Ejemplo:

**x = 3**

x contiene el valor 3

- **Su contenido puede modificarse** durante la ejecución de un programa.
- En Python las variables **no se declaran**. Simplemente, se usan.

# Variable

- El nombre de las variables pueden contener letras, dígitos y "_". Pero deben comenzar con una letra.

MiVariable
MiVar1
mi_var1
MiVar1
mi_var1

**¡¡Correctos!!**

"miVar"
mi_va**
1MiVariable
1MiVariable
"miVar"
mi_va**

**¡¡Incorrectos!!**

**Importante:**

En Python **HAY** diferencia entre mayúsculas y minúsculas: la variable **miVar** es distinta de la variable **MiVar.**

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

# Variable

```
print (‘ingrese un nombre’)
name=input() #  name es una variable string

print (‘ingrese una edad’)
edad=int(input()) # edad es una variable entera

edad=int(input(‘ingrese una edad’))
```

- El input siempre ingresa una cadena de caracteres. Si lo ingresado es un valor numérico, se debe hacer la conversión correspondiente.

# ¿Qué es un tipo de datos?

## Definición:

Un Tipo de datos define un **conjunto de valores** y las **operaciones válidas que pueden realizarse** sobre esos valores

- Conjunto de valores:
  - Representa **todos los valores posibles** que puede llegar a tomar una variable de ese tipo
- Operaciones permitidas:
  - Establece **qué operaciones son válidas** para los datos pertenecientes a dicho tipo

# Tipos Básicos - Enteros

Al asignar un número entero a una variable, Python le asociará el tipo **“int”.**

**Representa números enteros (positivos, negativos o cero).**

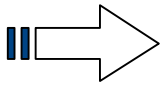
**No tiene un tamaño fijo (a diferencia de otros lenguajes como C o Java).**

**Puede crecer tanto como lo permita la memoria de la computadora.**

# Tipos Básicos - Reales

- Permite trabajar con valores con coma decimal.
- Se representan mediante el tipo **float**.
- Se almacenan en 64 bits.
- El rango de valores es de:
  - $\pm 2225073858507202_0 \times 10^{-308}$ a
  - $\pm 1797693134862315_7 \times 10^{308}$

```
>>> var_real1= 0.2703
>>> var_real2= 0.1e-3
```



**Notación científica.**

**$0.1 \times 10^{-3} = 0.1 \times 0.001 = 0.0001$**

Para el caso de necesitar representar fracciones de forma más precisa, se cuenta con el tipo **decimal**, desde la versión 2.4 de Python

# Operadores aritméticos

- Operaciones que pueden hacerse sobre variables numéricas y números.

| | Operador | Descripción |
|---|---|---|
| Operadores aritméticos | + | Suma |
| | - | Resta |
| | * | Multiplicación |
| | / | División |
| | - | Negación |
| | ** | Exponente |
| | // | División entera |
| | % | Resto de la división |

# Tipos Básicos - Booleanos

- Se los utiliza para indicar valores de verdad
- Permiten dos únicos valores:
  - True
  - False
- **Operadores lógicos: and, or, not**

| **and** | True | False |
|---|---|---|
| True | **True** | False |
| False | False | False |

| **or** | True | False |
|---|---|---|
| True | True | True |
| False | True | **False** |

| **not** | |
|---|---|
| True | False |
| False | True |

# Tipos Básicos - Booleanos

- **Operadores relacionales: ==, !=, <, <=, >, >=**
- **x=2; y=3**

| Operador | Descripción | Ejemplo | Resultado |
|---|---|---|---|
| = = | ¿x es igual a y? | x= =y | False |
| != | ¿x es distinto a y? | x!=y | True |
| < | ¿x es menor que y? | x<y | True |
| > | ¿x es mayor que y? | x>y | False |
| <= | ¿x es menor o igual que y? | x<=y | True |
| >= | ¿x es mayor o igual que y? | x>=y | False |

# Tipos Básicos - String o Cadenas

- Secuencia de caracteres (letras, números, marcas de puntuación, etc.)
- Se encierran entre comillas simples **' '** o dobles **“ ”**
- Algunos operadores: **+ Concatenación**
  *** Repetición**

```
>>> nombre='Pepe '
>>> apellido="Lopez"
>>> nombre + apellido
'Pepe Lopez'
>>> 'Lopez'*5
'LopezLopezLopezLopezLopez'
```

# Tipos Básicos - Cadenas

- Operadores de comparación: **==**, **!=**, **>**, **<**, **>=**,**<=**

  Ejemplos:

```
>>>    'pepe' == 'pepe'
True
>>> “juan”<”ana”
False
```

- Python utiliza un criterio de comparación de cadenas muy natural: **el órden alfabético**
- Python utiliza los **códigos ASCII** de los caracteres para **decidir** su **orden**

# Tipos Básicos - Cadenas

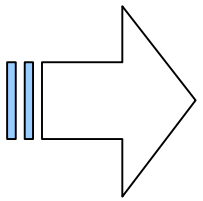
- Funciones predefinidas que manipulan cadenas:

| Funciones | Descripción | Ejemplo | Resultado |
|---|---|---|---|
| **int** | Convierte una cadena numérica a un valor entero | int(“123”) | 123 |
| **float** | Convierte una cadena numérica a un valor real | float(“123”) | 123.0 |
| **ord** | Devuelve el código ASCII de la cadena | ord(“a”) | 97 |
| **chr** | Devuelve el carácter correspondiente al valor ASCII | chr(89) | “T” |
| **str** | Convierte un valor entero a una cadena de caracteres | str(123) | “123” |

# Tipos Básicos - Cadenas

- Para saber el órden que ocupa un carácter se cuenta con las funciones predefinidas **“ord()”** y **“chr()”**, su función inversa.

```
>>>
>>> ord('a')
97
>>> chr(78)
'N'
>>>
```



Notar que:
'N'<'a'!!!

# Tipos Básicos - Cadenas

Otras cosas útiles.... Si **a** es una **variable** de tipo cadena (string)

| **Funciones** | Descripción | Ejemplo | Resultado |
|---|---|---|---|
| **a.lower()** | Convierte los caracteres de la cadena "a" en minúsculas | a="HOLA"<br>print(a.lower()) | "hola" |
| **a.upper()** | Convierte los caracteres de la cadena "a" en mayúsculas | a="hola"<br>print(a.upper()) | "HOLA" |

# Tipos Básicos - Cadenas

```
a.isupper()
a.islower()
a.isdecimal()   # si es nro y cadena no vacía
a.isalpha()      #  si es letra y cadena no vacía
a.isalnum()      #  número y letras, cadena no vacia
a.isspace()      # espacio
a.istitle()       #si es título, 1er letra may. resto minus.
a.startswith('s')
a.endswith('z')
m.split()          # retorna la lista de palabras que
                    #conforman la cadena m que están
                    #separadas por blancos
```

# Tipos Básicos - Cadenas

## Longitud de las cadenas

Uso de función predefinida

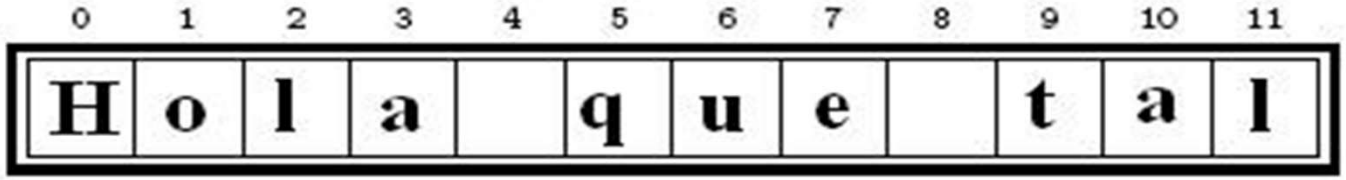
```
>>> cadena = 'Hola que tal'
>>> print('La longitud de la cadena es: ',len(cadena))
('La longitud de la cadena es: ', 12)
```

**len('')** devuelve longitud **0**

**len(' ')** devuelve longitud **1**

# Tipos Básicos - Cadenas

## Accediendo a los caracteres de las cadenas




```
>>> cadena = 'Hola que tal'
>>> print(cadena[0])
H
>>> print(cadena[5:8])
que
>>> print(cadena[9:-1])
ta
>>>
```

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

**[:]** devuelve toda la cadena

Índices negativos, recorren de **derecha a izquierda la cadena**

# Estructuras de Control Decisión

# Decisiones ….

**Sentencias condicionales**: Permiten comprobar condiciones y que el programa se comporte de una manera u otra, de acuerdo a esa condición.

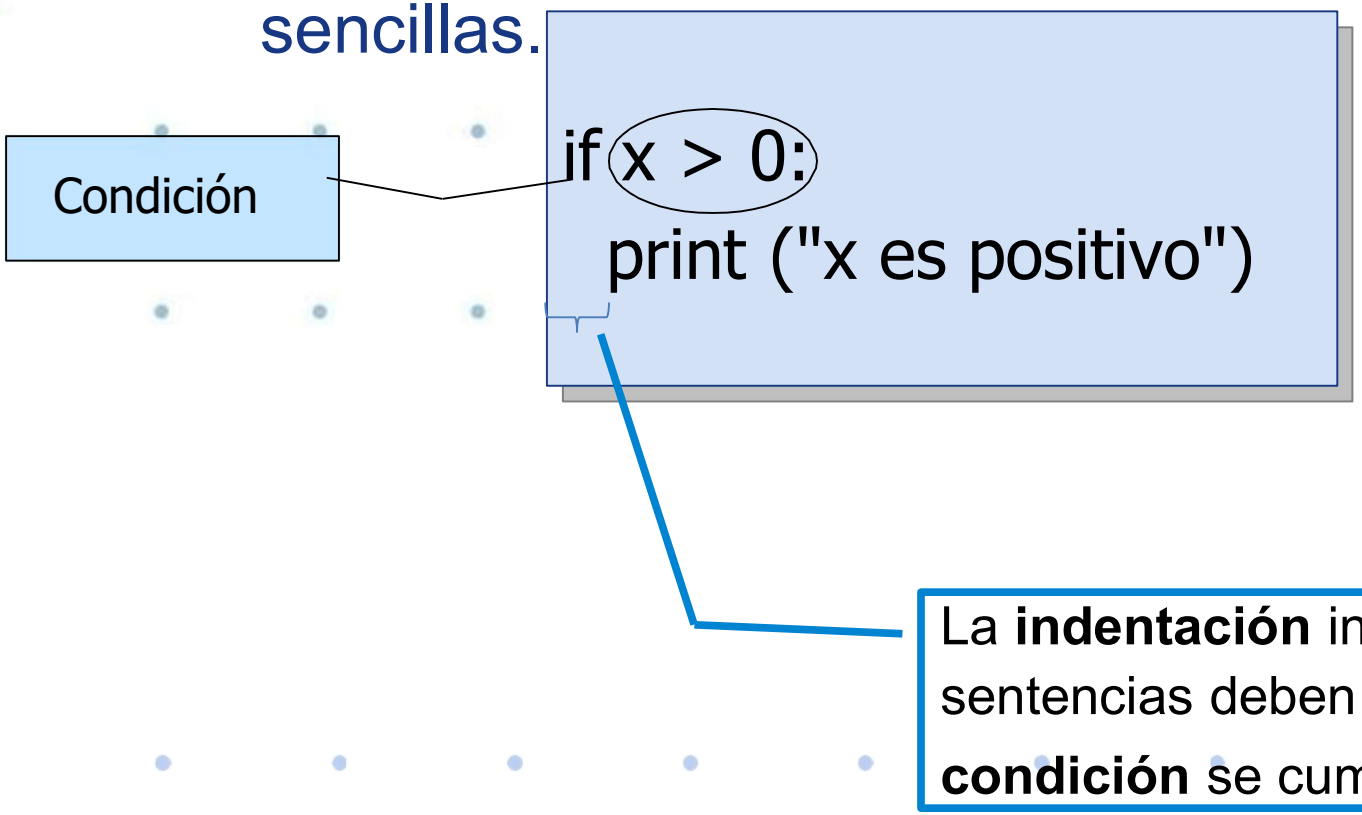
if

if .. else

if .. elif.. elif.. else

# Sentencia if

**Sentencia if**: Sentencia condicional más simple. Permite tomar decisiones sencillas.



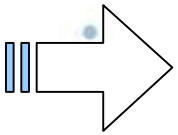
# Decisiones ….

## Sentencia if-else

- Permite establecer las acciones necesarias cuando la condición NO se cumple.

Ejemplo: Operador % (módulo)

Operador % (módulo)



```
if x%2==0:
    print (x, "es par")
else:
    print (x, "es impar")
```

# Decisiones ….

## Sentencia if-elif

¿Qué pasa cuando hay más de dos condiciones?

Ejemplo:

```
if edad>=0 and edad <2:
    print ("Bebe" )
elif edad >=2 and edad <13:
    print ("Niño/a" )
elif edad >=13 and edad <20:
    print ("Adolescente" )
else:
    print ("Adulto" )
```

Aquí aparecen varias condiciones excluyentes.

¿A qué valores referencia el else?

# Ejemplo 1

Escriba un programa que analice la edad ingresada por el usuario desde la consola. Si la edad es menor a 2 devuelve bebe; si es mayor que 2 y menor a 13 es niño/a; si es mayor a 13 y menor a 20 es adolescente y sino es adulto

# Ejemplo 1

```
edad = int(input("Ingrese edad: "))
if edad>=0 and edad<2:
 print “bebe”
elif edad>=2 and edad<13:
 print “niña/o”
elif edad>=13 and edad<20:
 print “adolescente”
else:
 print “adulto”
```