

Guía de Iniciación a la Programación con Python

Documentación Interna del Equipo de Desarrollo

Autor: Ander

1. ¿Qué es un condicional?

Un condicional es una estructura de control que permite al programa tomar decisiones basadas en si una condición es verdadera (True) o falsa (False). Es el equivalente a la lógica de "Si ocurre esto, haz aquello".

Sintaxis en Python

Se utiliza la palabra reservada `if`, seguida de la condición y dos puntos (:). El bloque de código que se ejecuta debe estar **indentado**.

Python

```
edad = 18
```

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

Tipos de condicionales

- **if**: Ejecuta un bloque si la condición se cumple.
 - **elif**: (Abreviatura de *else if*). Permite evaluar múltiples condiciones sucesivas.
 - **else**: Ejecuta un bloque si ninguna de las condiciones anteriores fue verdadera.
-

2. Bucles en Python: for y while

Los bucles son estructuras que permiten repetir un bloque de código varias veces. Son

fundamentales para evitar la redundancia y aplicar el principio **DRY** (*Don't Repeat Yourself*).

Tipos de Bucles

1. **Bucle for:** Se utiliza para iterar sobre una secuencia (una lista, una tupla, una cadena de texto o un rango). Es ideal cuando sabemos de antemano cuántas veces queremos repetir la acción.
 - *Ejemplo:* `for i in range(5): print(i)`
2. **Bucle while:** Se ejecuta mientras una condición específica sea verdadera. Es útil cuando no sabemos exactamente cuántas iteraciones serán necesarias.
 - *Ejemplo:* `while stock > 0: vender_producto()`

¿Por qué son útiles?

Permiten procesar grandes volúmenes de datos, automatizar tareas repetitivas y recorrer colecciones de elementos de manera eficiente sin escribir líneas de código adicionales de forma manual.

3. Listas por comprensión (List Comprehension)

Es una sintaxis elegante y concisa para crear nuevas listas a partir de un iterable existente en una sola línea de código. Sustituye la necesidad de crear una lista vacía y usar un bucle for tradicional.

Sintaxis

`nueva_lista = [expresion for elemento in iterable if condicion]`

Ejemplo comparativo:

Método tradicional:

Python

```
numeros = [1, 2, 3, 4]
cuadrados = []
for n in numeros:
    cuadrados.append(n ** 2)
```

Con lista por comprensión:

Python

```
cuadrados = [n ** 2 for n in numeros]
```

4. ¿Qué es un argumento en Python?

Un argumento es el valor que se envía a una función cuando esta es llamada. Los argumentos permiten que las funciones sean dinámicas y procesen diferentes datos según la necesidad.

Tipos comunes:

- **Argumentos posicionales:** Se asignan según el orden en que se pasan.
- **Argumentos de palabra clave (keyword arguments):** Se pasan especificando el nombre del parámetro (ej. nombre="Ander").

Ejemplo:

Python

```
def saludar(nombre): # 'nombre' es el parámetro
    print(f"Hola, {nombre}")
```

```
saludar("Carlos") # "Carlos" es el argumento
```

5. Funciones Lambda

Las funciones Lambda, también conocidas como **funciones anónimas**, son funciones de una sola línea que no tienen nombre. Se definen con la palabra clave lambda.

Sintaxis

lambda argumentos: expresion

Para qué se utilizan:

Se emplean para tareas rápidas y eficientes donde no queremos definir una función completa con def. Son muy comunes al trabajar con funciones de orden superior como filter(), map() o sorted().

Python

```
# Función que suma dos números  
suma = lambda a, b: a + b  
print(suma(5, 3)) # Resultado: 8
```

6. ¿Qué es un paquete PIP?

PIP (*Pip Installs Packages*) es el sistema de gestión de paquetes estándar para Python. Un "paquete" es esencialmente una colección de módulos y herramientas desarrolladas por otros programadores que podemos "instalar" en nuestro proyecto para no tener que reinventar la rueda.

Funciones principales:

- **Instalación:** pip install nombre_del_paquete (ej. pip install pandas).
- **Gestión:** Permite descargar librerías desde el repositorio oficial **PyPI** (Python Package Index).
- **Listado:** pip list muestra todos los paquetes instalados en el entorno actual.