## **Operadores Lógicos en Python**

Los operadores lógicos nos permiten combinar múltiples condiciones booleanas (verdaderas o falsas) para crear expresiones más sofisticadas. Los principales son:

* **and:** Devuelve True si ambas condiciones son True.
* Ejemplo: if edad >= 18 and edad <= 65:
* **or:** Devuelve True si al menos una de las condiciones es True.
* Ejemplo: if color == "rojo" or color == "azul":
* **not:** Niega una condición. Si la condición es True, not la convierte en False, y viceversa.
* Ejemplo: if not esta\_llovendo:

### **Ejemplos Prácticos**

Imaginemos que queremos escribir un programa que determine si una persona puede jubilarse. Consideraremos que la edad de jubilación es entre 60 y 65 años, pero si la persona ha contribuido más de 30 años al sistema de pensiones, podrá jubilarse a los 55 años.

Python

edad = 58 anos\_cotizados = 32if (edad >= 60 and edad <= 65) or (edad >= 55 and anos\_cotizados > 30): print("Puedes jubilarte.") else: print("Aún no puedes jubilarte.")

Use code [with caution.](file:///C:/faq#coding)

En este ejemplo:

* La primera parte de la condición (edad >= 60 and edad <= 65) verifica si la edad está dentro del rango estándar de jubilación.
* La segunda parte (edad >= 55 and anos\_cotizados > 30) verifica si la persona cumple con los requisitos de jubilación anticipada.

### **Combinando Operadores Lógicos**

Podemos combinar múltiples operadores lógicos para crear condiciones aún más complejas. Por ejemplo:

Python

if not (edad < 18 or edad > 65): print("Eres un adulto en edad laboral.")

Use code [with caution.](file:///C:/faq#coding)

Esta condición verifica si la edad no está por debajo de 18 ni por encima de 65, lo que significa que la persona es un adulto en edad laboral.

### **Tabla de Verdad de los Operadores Lógicos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **A and B** | **A or B** | **not A** |
| True | True | True | True | False |
| True | False | False | True | False |
| False | True | False | True | True |
| False | False | False | False | True |

Export to Sheets

[1. docplayer.gr](https://docplayer.gr/1640089-Simeioseis-gia-tin-pascal.html)

[docplayer.gr](https://docplayer.gr/1640089-Simeioseis-gia-tin-pascal.html)

### **Consejos Adicionales**

* **Indentación:** La indentación es crucial en Python para definir los bloques de código asociados a cada condición.
* **Paréntesis:** Utiliza paréntesis para agrupar condiciones y mejorar la legibilidad, especialmente cuando combinas múltiples operadores.
* **Simplificación:** A veces, puedes simplificar expresiones lógicas utilizando la ley de De Morgan o otras técnicas de lógica booleana.