## ¿Qué es un condicional?

Un condicional es una estructura de control en programación que permite ejecutar ciertas acciones basadas en una condición o conjunto de condiciones. Básicamente, un condicional evalúa una expresión booleana (es decir, una expresión que puede ser verdadera o falsa) y ejecuta un bloque de código dependiendo del resultado de esta evaluación.
Los condicionales son instrucciones que se le dan a un programa para que ejecute un bloque de código u otro si se cumplen determinados requisitos. De este modo, puede adaptarse a distintas situaciones e incluso responder ante usos o consultas inusuales.

El software entiende cada condición como verdadera o falsa, y actúa según dicha evaluación. Para explicar las circunstancias que se valoran, se hace uso de los operadores de Python, especialmente de los lógicos–cuando hay que comprobar más de un requisito a la vez–, y los de comparación –cuando la condición es única–.

Los condicionales suelen usarse junto a otros elementos, como los

operadores lógicos y los de comparación

Gracias a los condicionales que hay en Python, las plataformas pueden ajustarse a la interacción del/la usuario/a,controlar que cumple los requerimientos para registrarse – la longitud de una contraseña, por ejemplo– o que los datos de un/a cliente/a potencial son válidos y pueden entrar en el base de datos como lead. –¿el teléfono tiene la cantidad de números correcta?, ¿y el código postal?–. Para las operaciones más complicadas, se pueden combinar varios condicionales, como veremos en los siguientes epígrafes.

# Uso de if, else y elif en Python

Los principales condicionales en Python son **if, else y elif**.
“Traducidos” al lenguaje humano, serían:

| Condicionales principales | | |
|---|---|---|
| *Condicional* | *Significado* | ejemplo |
| if | si | edad = 18<br>if edad < 18:<br><br>print(“No puedes beber alcohol”.) |
| else | “si no” | edad = 18<br>if edad < 18:<br><br>print(“No puedes beber alcohol”.)<br><br>else:<br><br>print(“Puedes beber alcohol”.) |

| | | |
|---|---|---|
| elif (else + if) | “Si no, si” | edad = 18<br>if edad > 18:<br><br>print(“Puedes beber alcohol”.)<br><br>elif edad == 18:<br><br>print(“Como acabas de cumplir la mayoría de edad, puedes beber alcohol”.)<br><br>else:<br><br>print(“No puedes beber alcohol”.) |

Como se deduce a partir de la tabla anterior, cada uno de los condicionales en Python **marca una ruta a seguir:**

- **if:** ejecuta el código al que va asociado si la condición es verdadera. En el ejemplo, sería mostrar el mensaje “No puedes beber alcohol” si el/la usuario/a tiene menos de 18 años.
- **Else:** va después de un if. Aplica el código al que está conectado si la condición que el if marca es falsa. Es decir, si A no se cumple, else B.
- **elif:** si hay varias condiciones que comprobar antes de “decidir”, se utiliza if en la primera y elif en las siguientes. En la tabla, expresa que, si el/la usuario/a no es mayor de 18 años, sino que tiene justo los 18, puede beber alcohol legalmente.

# Condicionales anidados y operadores lógicos en Python

Los condicionales anidados en Python son el resultado de**introducir un condicional dentro de otro**. Como las matrioshkas, pero con requisitos en lugar de muñecas.

Como ocurre con el elif, se emplean para**c omprobar distintas variables**. Entonces, ¿qué los distingue exactamente de esta secuencia?

Diferencia entre elif y los condicionales anidados

| *elif* | *Condicionales anidados* |
| --- | --- |
| • Solo valora el requisito al que va asociado si los anteriores resultan ser falsos | • Valoran cada requisito independientemente del resto |
| • Simplifica el código y lo hace más fácil de leer y de comprender | • Permiten mayor precisión en las casuísticas complicadas |

Otra de las dudas que pueden surgirte es si los condicionales anidados suponen una **alternativa válida al operador** or. Lo cierto es que ambos hacen lo mismo: determinar si, como mínimo, uno de los requisitos expresados es verdadero. Por ejemplo, estos dos códigos indican las mismas instrucciones:

Condicionales anidados

```
x= 10
if x > 5:
  if x < 15:
    print(“x está entre 5 y 15.”)
  else:
    print(“x es mayor que 15.”)
else:
  print(“x es menor o igual a 5.”)
```

Operador or

```
x = 10
if x > 5 or x > 15:
  if x < 15:
    print(“x está entre 5 y 15.”)
  else:
    print(“x es mayor que 15.”)
else:
  print(“x es menor o igual a 5.”)
```

Hay otros operadores lógicos que pueden sustituir las anidaciones condicionales, como el **and**. Emplear unos y otros depende del grado de detalle que quieras darle al código y de lo **complejidad de las condiciones** que haya que describir. ¡Te recomendamos **practicar con**

ejercicios de Python para ir puliendo tu capacidad de decisión entre unos y otros!

## Prácticas recomendadas y errores comunes en el uso de condicionales

Para **usar correctamente los condicionales** en Python hay que seguir las buenas prácticas que recomiendan los/as desarrolladores/as. Estas son:

- Explicar los condicionales más complejos con **comentarios**, para que otros/as programadores/as puedan comprenderlo.
- Colocar las condiciones en la **secuencia correcta**, sobre todo si se usan sentencias como el elif, que valora los requisitos según el orden en el que se hayan escrito.
- Optar siempre por la **opción más sencilla.**Por ejemplo, usar el and cuando se pueda, en lugar de varios if anidados.
- **Recurrir a los paréntesis** cuando se combinan distintos condicionales y operadores.

Pese a que no seguir estos criterios es ya de por sí un **error**, al usar los condicionales en Python suelen cometerse otros como:

- **No escribir los dos puntos “:”** después de los condicionales.
- **Abusar de las anidaciones**, que hacen que el código sea cada vez más complejo y que el software esté menos optimizado.
- **No contemplar todos los requisitos** posibles.
- Realizar una **indentación incorrecta** o confundir los operadores.