ISPC
INSTITUTO SUPERIOR POLITÉCNICO CÓRDOBA

ESTRUCTURAS ALTERNATIVAS O CONDICIONALES

Existen ocasiones donde queremos repetir un código tantas veces como sea necesario para resolver un problema dado. Con el flujo de repetición solucionamos esta problemática mediante el uso de ciclos o bucles.

Un condicional o, mejor dicho, una sentencia o estructura condicional, es un mecanismo que **nos permite decidir si ejecutamos un determinado bloque o fragmento de código o no**, ya sea una sola vez (condicional simple) o varias veces (bucle o ciclo).

Las condicionales nos permiten por tanto, **crear distintas ramas de código que se ejecutarán o no en función de distintas condiciones que definamos**. Además, nos permiten decidir si ejecutamos un fragmento de código de manera repetida un número concreto de veces, es decir, un bucle.

Las condiciones en Python, se pueden utilizar de varias formas, más comúnmente es la instrucción **if**, el **else** es opcional. La palabra clave **elif** es la abreviatura de **else if**, y es útil para evitar el sangrado excesivo. Un if… elif … elif … es un sustituto de switch o case.

La sintaxis básica de la sentencia if

Una sentencia **if** en Python se describe como:

"Si la expresión evaluada, resulta ser verdadera (True), entonces ejecuta una vez el código en la expresión. Si sucede el caso contrario y la expresión es falsa, entonces No ejecutes el código que sigue."

La sintaxis general para la sentencia **if** básica es la siguiente:
if condicion:
ejecutar sentencia

Una sentencia **if** consiste en:

- La palabra reservada **if**, da inicio al condicional **if** .
- La siguiente parte es la condición, ésta puede evaluar si la declaración es verdadera o falsa. En Python estas son definidas por las palabras reservadas (True or False).
- Paréntesis (()) Los paréntesis son opcionales, no obstante, ayudan a mejorar la legibilidad del código cuando más de una condición está presente.
- Dos puntos: cuya función es separar la condición de la declaración de ejecución siguiente.

- Una nueva línea.
- Un nivel de **indentación** de cuatro espacios, que es una convención en Python. El nivel de **indentación** es asociado con la estructura de la declaración que sigue.
- Finalmente, la estructura de la sentencia, éste es el código que será ejecutado, únicamente si la sentencia a ser evaluada es verdadera. Es posible tener múltiples líneas en la estructura de código que pueden ser ejecutadas; en este caso es necesario tener cautela en cuanto a que todas las líneas tengan el mismo nivel de indentación.

A modo de ejemplo:

```
a = 1
b = 2

if b > a:
    print(" b es mayor que a")
```

¿Cómo funcionan las sentencias if..else ?

Cómo mencionamos anteriormente, una sentencia **if** ejecuta el código, solo en caso de cumplirse la condición estipulada.

¿Pero qué sucede si queremos ejecutar el código alternativo en caso de no cumplirse las condiciones?

Es en este caso que es necesario usar **else**.
La sintaxis de una sentencia **if..else** es parecida a la siguiente:

if condicion:
*ejecutar codigo si la condicion es **True***
else:
*ejecutar codigo si la condicion es **False***

Una sentencia **if..else** en Python significa:

*"Cuando la expresión **if** se evalúa como **True**, entonces ejecuta el código que le sigue. Pero si se evalúa como **False**, entonces ejecuta el código que sigue después de la sentencia **else.**"*

La sentencia **else** está escrito en una nueva línea, posterior a la última línea del código indentado, y no puede ser escrita por sí misma.

- Una sentencia **else** tiene como prerrequisito una sentencia **if,** siendo a la vez, parte de él.
- El siguiente código también necesita ser indentado **4** espacios para definirlo como parte de la cláusula **else**.
- El código posterior a la sentencia condicional **else** se ejecuta *sí y solo sí* la primera parte, del condicional **if** se evalúa como **False**. Si tu sentencia en la condicional **if** se evalúa como **True**, será ese bloque de código el que se ejecutará y el código que sigue a **else** no se ejecutará nunca.

El bloque **else** se ejecuta cuando:

```
a = 1
b = 2

if a > b:
    print(" a es mayor que b")
else:
    print("b es mayor que a")
```

Obteniendo como resultado:

```
b es mayor que a
```

¿Cómo funciona elif en Python?

¿Qué si necesitamos más de dos opciones?

En vez de decir:

"Si la primera condición es verdadera, realiza esto, si no, realiza esto otro", ahora le indicamos al programa,

"Si esto no es verdadero, intenta esto otro, y si todas las condiciones fallan en ser verdaderas, entonces haz esto.

elif es lo que buscamos.

La sintaxis básica es la siguiente:

```
if primera_condicion:
    ejecutar sentencia
elif segunda_condicion:
    ejecutar sentencia
else:
    ejecutar sentencia alternativa si todas las condiciones previas son son evaluada
```

Podemos usar más de un **elif**. De esta forma tenemos más condiciones y más opciones.

Por ejemplo:

```
x = 1

if x > 10:
    print(" x es mayor que 10!")
elif x < 10:
     print("x es menor que 10!")
elif x < 20 :
     print("x es menor que 20!")
else:
    print("x es igual a 10")
```

Resultado:

```
x es menor que 10!
```

En este ejemplo, **if** pone a prueba una condición específica, los bloques **elif** son dos alternativas, y el bloque final **else** es la solución final cuando todas las condiciones previas no se han cumplido.

Nota: Ten cuidado del orden en el que escribes las sentencias **elif**.

Dependiendo del contexto, las sentencias condicionales pueden volverse más avanzados y complejos.

Las sentencias condicionales pueden ser anidados dentro de otras sentencias condicionales, dependiendo de las características del problema que intentes resolver, y de la lógica detrás de su solución.