Unidad 3

Estructuras repetitivas

Hasta el momento vimos que en Python el código se ejecuta de forma secuencial, es decir, se ejecuta línea por línea hasta que el intérprete no tiene más código para procesar. Sin embargo, existen casos donde vamos a necesitar que un bloque de código se repita.

El proceso de repetir bloques de código se conoce como **iteración**, **las estructuras repetitivas** (también conocidas como bucles) **se utilizan para que los programas implementen iteraciones.**

Bucle while

En Python el bucle while se utiliza para ejecutar un bloque de código mientras se cumpla una condición.

La sintaxis tiene el siguiente formato:

```
while condicion:
    #Bloque de codigo
```

Donde por ejemplo si quisiéramos que se muestre por pantalla los números de 1 hasta 5 podríamos escribir el siguiente programa:

```
contador = 1
while contador <= 5:
    print("Contador:", contador)
    contador += 1
```

Si ejecutamos el programa veremos por pantalla:

```
Contador: 1
Contador: 2
Contador: 3
Contador: 4
Contador: 5
```

En este ejemplo se ejecuta el bloque de código mientras contador sea menor o igual a 5.

Tener en cuenta lo siguiente:

- La **condición** se evalúa en cada **iteración**
- Si la **condición** nunca se vuelve falsa, el **bucle** puede ser **infinito**
- Se puede utilizar **break** para salir del bucle o **continue** para saltar a la siguiente **iteración**
- Podemos agregar un bloque de código cuando finalicen las iteraciones utilizando la cláusula **else**

Vemos un ejemplo utilizando break para salir de un bucle. Siguiendo el ejemplo anterior podríamos hacer un programa que cuente desde 1 a 5 pero cuando llegue a 3 debe salir del bucle e informarlo por pantalla.

```
contador = 1
while contador <= 5:
    print("Contador:", contador)
    contador += 1
    if contador == 3:
        print("¡finalizó el bucle!")
        break
```

Si ejecutamos el programa veremos por pantalla:

```
Contador: 1
Contador: 2
¡finalizó el bucle!
```

En este ejemplo el bucle se ejecuta hasta que el contador es igual a 3.

Ahora veamos un ejemplo utilizando continue para saltarnos la iteración. Siguiendo los ejemplos anteriores, vamos a crear un programa que cuente desde 1 a 5 pero cuando llegue a 3 debe saltarse la iteración y seguir contando hasta 5.

```
contador = 1
while contador <= 5:
    if contador == 3:
        print("¡Se salto la iteración!")
        contador += 1
        continue
    else:
        print("Contador:", contador)
        contador += 1
```

Si ejecutamos el programa veremos por pantalla:

```
Contador: 1
Contador: 2
¡Se salto la iteración!
Contador: 4
Contador: 5
```

En este ejemplo el bucle se ejecuta hasta que el contador es igual a 5 pero salta la iteración cuando el contador es igual a 3.

Por último, también podremos ejecutar un bloque de código cuando finalicen las iteraciones utilizando la cláusula else.

Ejemplo:

```
contador = 1
while contador <= 5:
    print("Contador:", contador)
    contador += 1
else:
    print("El bucle while finalizó")
```

Si ejecutamos el programa veremos por pantalla:

```
Contador: 1
Contador: 2
Contador: 3
Contador: 4
Contador: 5
El bucle while finalizó
```

En este caso cuando la condición se vuelve falsa (cuando contador es 6) se ejecuta el bloque de código de la cláusula else.

Contadores y Acumuladores

Anteriormente vimos que la variable contador incrementaba su valor en cada iteración y luego la utilizábamos para determinar si el bucle debía seguir ejecutándose. Este tipo de variables se conoce como contadores y son importantes dentro de las estructuras repetitivas, como asi también las variables de tipo acumulador.

Sus características pueden resumirse de la siguiente manera:

Contador:

- Se usa para contar cuántas veces ocurre algo.
- En general se incrementa en cada iteración.
- Ejemplo típico: contador += 1

Acumulador:

- Se usa para sumar (o acumular) valores en cada iteración.
- Ejemplo típico: acumulador += valor

¿Por qué se usan con bucles?

- Porque necesitamos llevar un control de lo que sucede en cada iteración
- Para contar las iteraciones de un bucle
- Para almacenar valores que se ingresan o se calculan
- Calcular promedios, totales, estadísticas, etc.

Veamos un ejemplo donde un programa cuenta los números desde 1 hasta 5 y luego muestra por pantalla la suma de todos ellos.

```
contador = 1
acumulador = 0
while contador <= 5:
    print("Contador:", contador)
    acumulador += contador # 1 + 2 + 3 + 4 + 5 = 15
    contador += 1
else:
    print("La suma de todos los valores es: ", acumulador)
```

```
Contador: 1
Contador: 2
Contador: 3
Contador: 4
Contador: 5
La suma de todos los valores es:  15
```

En este caso el contador se utiliza para controlar cuando debe finalizar el bucle y el acumulador lo utilizamos para almacenar el valor del contador en cada iteración.