# **Bucle For.**

Repetir de una forma controlada la ejecución de un codigo, son sentencias que nos permiten iterar un elemento.

No todos los elementos se pueden iterar, solo los elementos iterables, o sea, un elemento iterable es un elemento que tiene un iterador que define como se va iterar, un elemento es iterable cuando tiene algo que defina como se va iterar, o sea si no tiene nada que recorrer, porque eso es iterar, ir recorriendo el elemento en pedazos.

Las listas son iterables, las tuplas tambien, los diccionarios, conjuntos y cadenas de texto.

En Python los bucles son “For in”, que es un bucle que crea una variable que en cada vuelta va a ser un pedazo de esa variable.

Veamos un mejor ejemplo.

Supongamos que tenemos una lista de dinosaurios

dinosaurios = ["Triceratops", "Raptor", "Spinosaurio", "Gallinimus"]

Ahora como iteramos esta lista con “For”, primero decimos

for dinosaurio in dinosaurios:

Ahora podemos ejecutar el código, la pregunta será ¿a que va ser igual dinosaurio?.

La primera vez que se ejecute el código dinosaurio será igual al primer elemento o sea a Triceratops, esto lo podemos comprobar con un print. Cuando termine de dar la vuelta al bloque, volverá a comprobar los elementos de la lista si hay mas elementos entonces se volverá a ejecutar, pero en esta segunda vuelta será el segundo elemento, osea Raptor, y vuelve a hacer lo mismo hasta que ya no haya mas elementos, y pasamos a la siguiente función.

dinosaurios = ["Triceratops", "Raptor", "Spinosaurio", "Gallinimus"]

for dinosaurio in dinosaurios:

    print(dinosaurio)



Otro ejemplo con el bucle For pero con números.

Digamos tenemos una lista con números, pero a cada uno de esos números los queremos multiplicar por 2.

numeros = [23, 10, 2, 54, 231, 1]

for numero in numeros:

    resultado = numero \* 2

    print(resultado)



De esta manera podemos recorrer las listas, o sea iterarlas.

Podemos iterar dos listas usando For anidados o sea un for dentro de otro for, o usando una función en particular que es “zip()”, veamos como.

Las listas deben tener la misma cantidad de elementos.

dinosaurios = ["Triceratops", "Raptor", "Spinosaurio", "Gallinimus"]

numeros = [23, 10, 2, 54]

Ahora debemos crear el bucle de esta manera:

for numero, dinosaurio in zip(dinosaurios, numeros):

    print(f"Recorriendo lista 1: {dinosaurio}")

    print(f"Recorriendo lista 2: {numero}")

Usando la función zip, vemos como podemos recorrer en este caso las listas.



Aquí aparece primero el numero en vez del animal por como lo ordenamos en el zip. Podemos iterar mas listas pero deben tener el mismo tamaño.

Pero todo es al mismo tiempo, no uno por uno.

## **Range.**

Otra forma de iterar es usando la función “range”.

for numero in range(5, 10):

    print(numero)

En este caso vemos que definimos la variable numero, y usando la función range definimos los rangos de números que queremos que se almacenen en el bucle.



Siendo el 5 el primero parámetro desde donde inicia, y el 10 siendo en donde termina, pero el ultimo no cuenta.

Y sino definimos un segundo parámetro inicia desde el 0 hasta el numero que le digamos.

for numero in range(5):

    print(numero)



Otra manera de aplicarlo con la lista de números creada anteriormente podría ser esta.

numeros = [23, 10, 2, 54]

for numero in range(len(numeros)):

    print(numeros[numero])

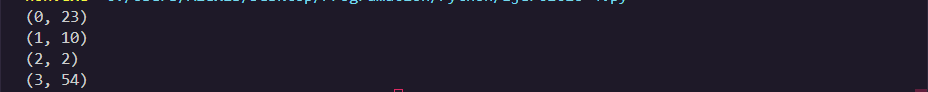


Lo que estamos diciendo aquí es que el rango es la cantidad de elementos de nuestra lista que es 4, pero las listas iniciando desde el 0, y despues imprimimos en pantalla nuestra lista, pero mostrando cada numero de ella. Pero es una forma no optima de realizarlo.

La forma correcta seria con la función “enumerate”:

for numero in enumerate(numeros):

    print(numero)



Aquí al usar la función enumerate, nos enumera los elementos de la lista con su índice, numero es una tupla

Y podemos definir en variables el índice y el valor de esta manera:

for numero in enumerate(numeros):

    indice = numero[0]

    valor = numero[1]

    print(f"El indice es: {indice} y el valor es: {valor}")



Pero una manera más practica y mas elegante de hacer este mismo proceso es hacer esto:

numeros = [23, 10, 2, 54]

for indice, numero in enumerate(numeros):

    print(f"El indice es: {indice} y el valor es: {numero}")



Aquí con mas números.

numeros = [23, 10, 2, 54, 123, 34, 354]

for indice, numero in enumerate(numeros):

    print(f"El indice es: {indice} y el valor es: {numero}")



## **Else en el for.**

Tambien podemos usar el “else” dentro del “for”.

dinosaurios = ["Triceratops", "Raptor", "Spinosaurio", "Gallinimus"]

numeros = [23, 10, 2, 54, 123, 34, 354]

for numero in numeros:

    print(f"Ejecutando el ultimo bucle, valor acutal {numero}")

else:

    print("El bucle termino")



Podemos hacer lo mismo con las tuplas.

Tambien con conjuntos a excepción de recorrer números de la forma no optima.

Pero los diccionarios funcionan diferente.

## **Iterar un diccionario.**

diccionario = {

    "Procesador" : '5600G',

    "Memoria RAM" : '16 GB',

    "Almacenamiento" : '2 TB'

}

for key in diccionario:

    print(key)



Aqui solo nos estará mostrando las claves ya que si especificamos en este caso “key” (puede ser otra cosa) nos mostrara las mismas, pero para hacer que nos muestre el resultado debemos de hacer lo siguiente:

diccionario = {

    "Procesador" : '5600G',

    "Memoria RAM" : '16 GB',

    "Almacenamiento" : '2 TB'

}

for key in diccionario.items():

    print(key)



Esto nos devuelve una tuple que contiene clave y valor del diccionario.

diccionario = {

    "Procesador" : '5600G',

    "Memoria RAM" : '16 GB',

    "Almacenamiento" : '2 TB'

}

for datos in diccionario.items():

    key = datos[0]

    value = datos[1]

    print(f"La clave es: {key} y el valor es: {value}")



De esta manera recorremos diccionarios. O una manera mas optima seria asi.

diccionario = {

    "Procesador" : '5600G',

    "Memoria RAM" : '16 GB',

    "Almacenamiento" : '2 TB'

}

for key, value in diccionario.items():

    print(f"La clave es: {key} y el value es: {value}")

Y nos dará lo mismo.

## **Mas iteraciones.**

Si queremos saltarnos un elemento de en este caso una lista, podemos agregar un if con un “continue”, de esta manera:

frutas = ["Manzana", "Piña", "Papaya", "Banana"]

for fruta in frutas:

    if fruta == 'Piña':

        continue

    print(f"Me voy a comer una {fruta}")



De esta manera podemos ver que saltamos el elemento Piña gracias al if, especificando que si fruta es igual a piña, continue el resto del bucle, pero saltándose ese mismo elemento.

Y si nosotros queremos forzar a terminar el bucle usamos la sentencia “break”:

for fruta in frutas:

    if fruta == 'Papaya':

        break

    print(f"Me voy a comer una {fruta}")



Con este decimos que si el bucle llega al elemento de Papaya, el bucle se termina se corta. Y continua con el resto del codigo, pero el bucle queda ahí terminado.

### **Iterar una cadena.**

cadena = "Me gusta el cocholate"

for letra in cadena:

    print(letra)



## **Iterar una sola linea de código**.

numeros = [2, 4, 6, 8]

numeros\_duplicados = [x \* 2 for x in numeros]

print(numeros\_duplicados)

Lo que estamos diciendo aquí es que primero que nada creamos nuestra lista de números, y despues creamos el bucle for dentro de la variable números\_duplicados, pero como valor de esta variable decimos, que x lo multiplicamos por 2 e iniciamos el bucle que nos dice que creamos a x en la lista de números, o sea cada valor de la lista números, se guarda en x y se multiplica por 2 para despues guardarse dentro de otra lista dentro de la variable números\_duplicados.

