Buenos Aires aprende
Agencia de Habilidades para el Futuro

Talento Tech

Iniciación con Python

Clase 09 - “Bucles for”

Buenos Aires Ciudad

¡Les damos la bienvenida!

 Vamos a comenzar a grabar la clase

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

Talento Tech

Clase 08.

Ruta de avance

1. Pre-Entrega del Proyecto Final Integrador.
2. Menú de opciones.
3. Pedir, procesar y mostrar datos.

Clase 09.

Bucle for

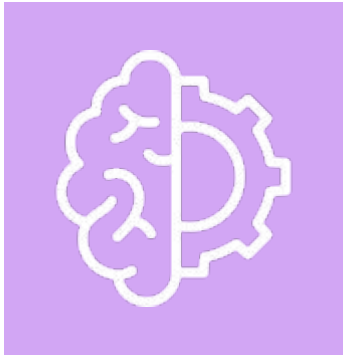
1. Control de flujo: bucles for
2. Slices de listas.

Clase 10.

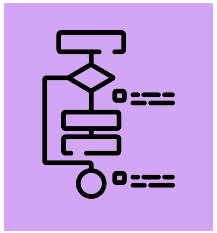
Diccionarios

1. Diccionarios: uso y métodos esenciales.
2. Uso de diccionarios como medio de almacenamiento temporal de datos

Pero antes...



¡Resolvamos los “**Ejercicios prácticos**” de la clase anterior!



Bucles for

Bucle for

El bucle for es una estructura de control que se utiliza para iterar sobre una secuencia de elementos, como una lista, cadena, rango o cualquier otro objeto iterable.

Su sintaxis es simple y compacta, y consta de tres partes: la palabra clave for, una variable de iteración que toma el valor de cada elemento en la secuencia en cada iteración, la palabra clave in, y la secuencia sobre la que se va a iterar.

```
for variable in secuencia:
    # Bloque de código
```

El bucle for recorre cada elemento de la secuencia y ejecuta un bloque de código para cada uno de ellos, lo que lo hace ideal para realizar tareas repetitivas o para procesar cada elemento de una colección.

Bucle for con listas

Con el bucle for no necesitas usar contadores, ya que la variable del ciclo puede asumir esa función. Esto permite escribir algunos programas de una manera más compacta.
En el ejemplo, cada repetición del bucle, la variable producto toma uno de los elementos de la lista, y lo muestra en pantalla.

```
productos = ["manzanas", "naranjas", "bananas", "peras"]

for producto in productos:
   print("Producto disponible:", producto)
```

```
Producto disponible: manzanas
Producto disponible: naranjas
Producto disponible: bananas
Producto disponible: peras
```

Bucle for con tuplas

Sabemos que una tupla es similar a una lista, pero con la diferencia de que es inmutable (no se puede modificar una vez creada). Al igual que con las listas y cadenas, podemos recorrer una tupla con for:

```
numeros = (1, 2, 3, 4, 5)

for numero in numeros:
   print("Número:", numero)
```

```
Número: 1
Número: 2
Número: 3
Número: 4
Número: 5
```

Bucle for con *strings*

Una cadena de texto en Python es una secuencia de caracteres. Esto significa que, usando for, podemos recorrerla carácter por carácter. Fíjate que fácil es:

```python
frase = "Python"

for letra in frase:
   print("Letra:", letra)
```

```
Letra: P
Letra: y
Letra: t
Letra: h
Letra: o
Letra: n
```

Bucle for con range()

Esta estructura se utiliza cuando sabemos la cantidad de repeticiones a efectuar. Tiene el siguiente formato:

```
for i in range(inicio, fin, paso):
    # sentencia1
    # sentencia2

# primer sentencia fuera del for
```

i: Variable que incrementa su valor en paso unidades en cada iteración.
inicio: Es el valor inicial de i
fin: El ciclo se repite mientras i sea menor que fin.
paso: Valor en que se incrementa i en cada iteración.

Bucle for con range()

range() puede tener uno, dos o tres parámetros:

- range(inicio, fin, paso)
- range(inicio, fin): Se asume que paso = 1.
- range(fin): Se asume que inicio = 0 y paso = 1.

La función range() no incluye el último valor en la secuencia. Por ejemplo, range(1, 6) generará los números del 1 al 5, pero no incluirá el 6 en la secuencia.

Ejemplo de uso:

```
for num in range(5, 16, 2):
    print(num, end=" ")
```

Salida:

```
5 7 9 11 13 15
```

```
for num in range(3, 11):
    print(num, end=" ")
```

```
3 4 5 6 7 8 9 10
```

```
for num in range(10):
    print(num, end=" ")
```

```
0 1 2 3 4 5 6 7 8 9
```

Break

Break permite salir de un bucle **for** (o **while**) en el momento que se cumpla alguna condición.

```
suma = 0

for cont in range(15):
    print(cont)
    suma = suma + cont
    if cont == 3:
        break

print("La suma es:", suma)
```

El uso de **break** no se considera una buena práctica. Si un bucle **for** requiere un break, quizás pueda resolverse el proceso usando en su lugar **while**.

```
0
1
2
3
La suma es: 6
```

Ejemplo 1: suma y promedio

Con el bucle **for** no necesitamos usar **contadores**, ya que la variable del ciclo puede asumir esa función. Esto permite escribir algunos programas de una manera más compacta.
Este programa permite ingresar 5 valores por teclado, obtener su suma y su promedio:

```
suma = 0
for cont in range(5):
   num= int(input("Ingrese un número: "))
   suma = suma + num

print('La suma es:',suma)
print('El promedio es:',suma/(cont+1))
```

```
Ingrese un número: 2
Ingrese un número: 6
Ingrese un número: 9
Ingrese un número: 15
Ingrese un número: 3
La suma es: 35
El promedio es: 7.0
```

Ejemplo 2: procesar datos

Este programa simula la carga de 5 importes de productos de un supermercado, mostrando el total al final.

```
total = 0

for i in range(5):
    importe = float(input("Ingrese el importe: "))
    total = total + importe

print("El total de los importes ingresados es:" ,total)
```

```
Ingrese el importe: 4
Ingrese el importe: 9
Ingrese el importe: 12
Ingrese el importe: 15
Ingrese el importe: 6
El total de los importes
ingresados es: 46.0
```

¡Vamos a la práctica! 

Ejercicios prácticos

Optativos | No entregables

Suma de números naturales

Desarrolla un programa en Python que calcule la suma de los números naturales hasta un número dado utilizando un bucle for. La suma de los números naturales hasta el número n se define como la suma de todos los números enteros positivos desde 1 hasta n.
Por ejemplo, la suma de los números naturales hasta 6 es 1 + 2 + 3 + 4 + 5 + 6 = 21.

Tips:

- ¡Usá range()!
- El programa debe pedir un número entero n y devolver la suma de los números naturales hasta n.

Ejercicios prácticos

Optativos | No entregables

Mostrar los códigos de productos ingresados

En un sistema de inventario, cada producto tiene un código que lo identifica. Escribí un programa que permita ingresar los códigos de 4 productos y luego mostrálos uno por uno, junto con su posición en la lista.

Ejemplo: si los códigos ingresados son "P001", "P002", "P003", "P004", el programa debe mostrar:

```
Producto 1: P001
Producto 2: P002
Producto 3: P003
Producto 4: P004
```

Tips:

- Utilizá un bucle for y range() para recorrer los códigos e imprimirlos.