

Python Básico

Prof. Ariel Palazzesi

Comenzamos a grabar la clase

Módulo 2:

Control de flujo

Clase 07

Listas y Tuplas I

1. Creación y manipulación de listas y Tuplas.
2. Diferencias entre listas y tuplas.
3. Métodos básicos de listas: `len()`, `append()`, `insert()`, `extend()`, `remove()`, `pop()`, `clear()`, `in`
4. Slices de listas, tuplas y cadenas.

Clase 08

Listas y Tuplas II

1. Listas y tuplas con `for` y `while`.
2. Uso de `list()`
3. Métodos avanzados de listas: `sort()`, `sorted()`, `index()`, `in`, `count()`.

Clase 09

Cadenas de caracteres

1. Manipulación de cadenas de caracteres.
2. Funciones y métodos comunes: `len()`, `*`, `split()`, `replace()`, `join()` e `in` f-strings.
- 3.

Listas II

Listas, tuplas y bucles

✓ Es posible recorrer una lista utilizando **for** y **range()** o **while** para acceder a los elementos de una lista a través de sus índices:

Código Python

```
# Definimos una lista de números
lista = [2, 3, 4, 5, 6]

# Sumamos sus elementos
suma = 0
for i in range(len(lista)):
    suma = suma + lista[i]
print(suma) # 20
```

Código Python

```
# Definimos una lista de números
lista = [2, 3, 4, 5, 6]
# Sumamos sus elementos
suma = 0
i = 0
while i < len(lista):
    suma = suma + lista[i]
    i = i + 1
print(suma) # 20
```

Listas con for



✓ **for** permite iterar en forma directa los elementos de una colección, como una lista o una tupla, *sin necesidad de generar la secuencia de subíndices*. En este caso la variable **numero** toma el elemento de la lista:

Código Python

```
lista = [2, 3, 4, 5, 6]

for numero in lista:
    suma = suma + numero
print(suma) # 20
```

Listas | max(), min() y sum()

- ✓ La función **max()** devuelve el mayor elemento de una lista.
- ✓ La función **min()** devuelve el menor elemento de una lista.
- ✓ La función **sum()** devuelve la suma de los elementos de una lista.

Código Python

```
lista = [3, 4, 5, 6]
print('El mayor elemento de la lista es:', max(lista))      # 6
print('El menor elemento de la lista es:', min(lista))      # 3
print('La suma de los elementos de la lista es:', sum(lista)) # 18
```

Listas | sort()



✓ El método **sort()** ordena los elementos de la lista en su lugar, de menor a mayor. Funciona con listas con todo tipo de datos, *y modifica la propia lista*.

Código Python

```
numeros = [3, 1, 4, 2, 5]
numeros.sort()
print('Lista de números ordenada:', numeros)

nombres = ["Juan", "Ana", "Laura", "Héctor"]
nombres.sort()
print('Lista de nombres ordenada:', nombres)
```


Listas | sorted()

✓ La función **sorted()** devuelve una lista con los elementos de la lista ordenados.
No modifica la lista original.

Código Python

```
numeros = [4, 2, 7, 1, 9]

# Devuelve una nueva lista, ordenada:
ordenada = sorted(numeros)

print(numeros) # Salida: [4, 2, 7, 1, 9]
print(ordenada) # Salida: [1, 2, 4, 7, 9]
```



Listas | index()

✓ El método **index()** se utiliza para encontrar la posición (índice) de la primera aparición de un elemento en una lista o cadena.

Se puede, de manera opcional, indicar desde qué y hasta cuál elemento realizar la búsqueda.

Código Python

```
numeros = [10, 20, 30, 40, 50]
indice = numeros.index(30)
print("El índice de 30 es:", indice)
# Salida: El índice de 30 es: 2
```

Código Python

```
numeros = [10, 20, 30, 20, 50]
indice = numeros.index(20, 2, 5)
print("El índice de 20 entre 2 y 5 es:", indice)
# Salida: El índice de 20 entre 2 y 5 es: 3
```

Listas | in y not in

✓ Los operadores **in** y **not in** recorren la lista de izquierda a derecha hasta encontrar una coincidencia (o llegar al final). Devuelven **True** o **False**, dependiendo si el elemento está o no en la lista.

Código Python

```
invitados = ["Ana", "Carlos", "Juan", "María"]

nombre = input("Ingresa tu nombre: ")

if nombre in invitados:
    print(";Bienvenido/a a la fiesta, " + nombre + "!")
else:
    print("Lo siento, " + nombre + ", no estás en la lista.")
```

Listas | count()

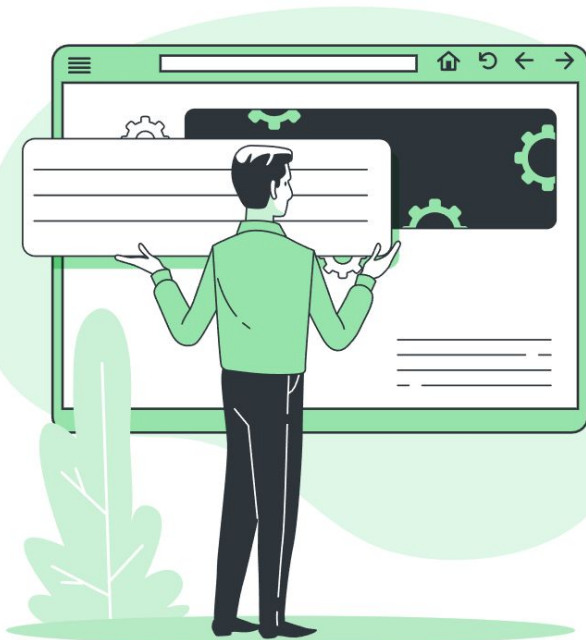
✓ El método **count()** se utiliza para contar cuántas veces aparece un elemento en una lista, cadena u otros iterables. Si el elemento no está presente, devuelve 0.

Código Python

```
numeros = [1, 2, 3, 2, 4, 2]
cantidad = numeros.count(2)
print("El número 2 aparece:", cantidad, "veces.")
# Salida: El número 2 aparece: 3 veces.
```



Listas | list()

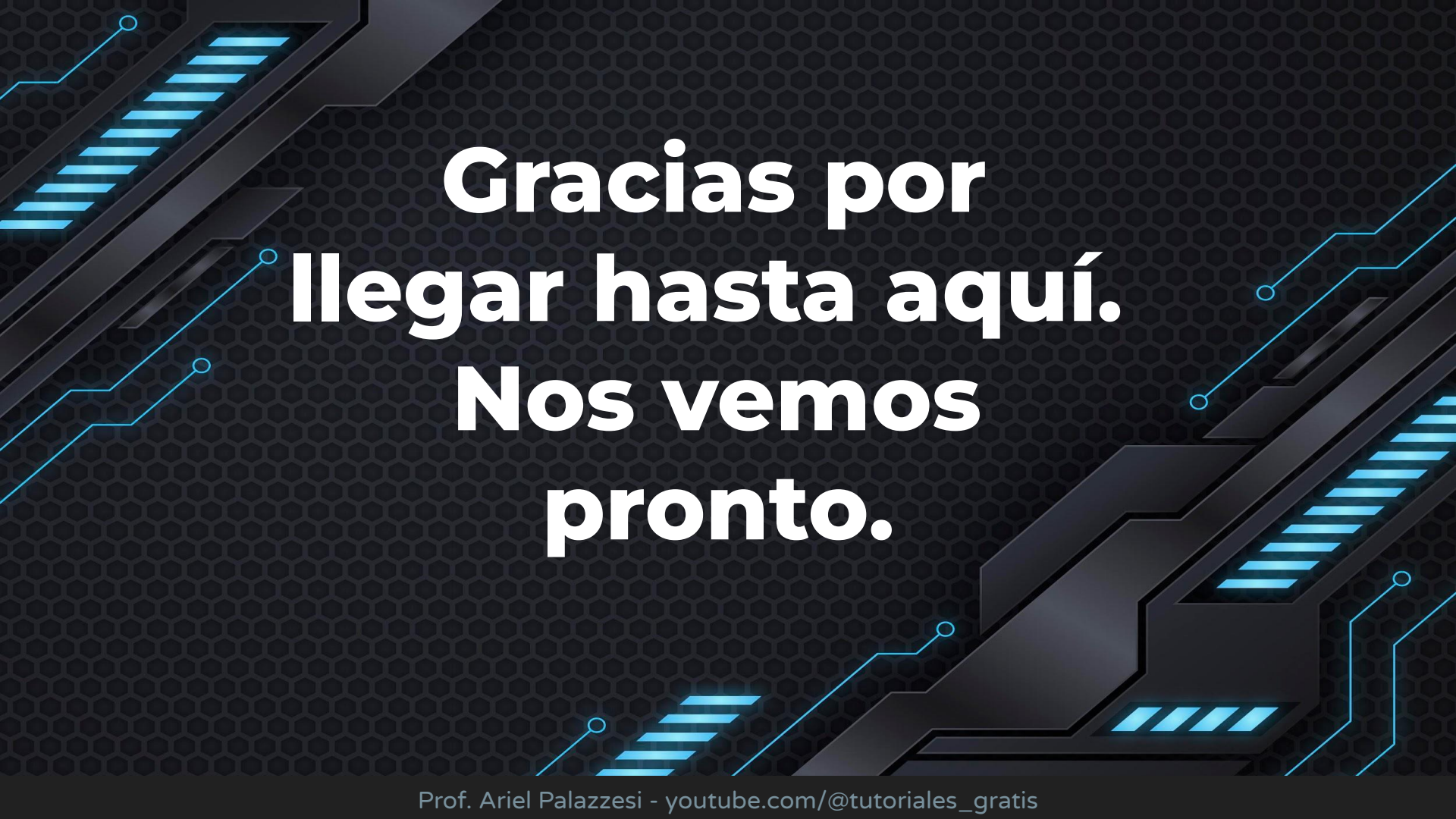


✓ El método **list()** se utiliza para crear una lista a partir de un iterable. Es una forma de convertir otros tipos de datos iterables en listas.

El iterable original permanece sin cambios; se crea una nueva lista.

Código Python

```
cadena = "Python"
lista = list(cadena)
print(lista)  # Salida: ['P', 'y', 't', 'h', 'o', 'n']
```



**Gracias por
llegar hasta aquí.
Nos vemos
pronto.**