Resumen de los aspectos más importantes de listas, tuplas y diccionarios en Python:

**Listas**

**1. Definición:**

- Colección ordenada y mutable.

- Permite elementos duplicados.

**2. Creación:**

python

lista = [1, 2, 3, 4, 5]

**3. Acceso a elementos:**

- Por índice: lista[0] (primer elemento), lista[-1] (último elemento).

- Slicing: lista[1:4] (sublista).

**4. Modificación:**

- Puedes cambiar el valor de un elemento: lista[0] = 10.

- Añadir elementos con append(): lista.append(6).

- Insertar elementos en una posición específica con insert(): lista.insert(1, 2.5).

- Eliminar elementos con remove(): lista.remove(2).

- Eliminar el último elemento con pop(): lista.pop().

**5. Métodos comunes:**

- extend(): Añade múltiples elementos: lista.extend([6, 7]).

- sort(): Ordena la lista in-place.

- reverse(): Invierte el orden de la lista.

**6. Uso:**

- Ideal para colecciones de elementos que puedan cambiar a lo largo del tiempo.

**Tuplas**

**1. Definición:**

- Colección ordenada e inmutable.

- Permite elementos duplicados.

**2. Creación:**

python

tupla = (1, 2, 3, 4, 5)

**3. Acceso a elementos:**

- Por índice: tupla[0], tupla[-1].

- Slicing: tupla[1:4].

**4. Modificación:**

- Las tuplas son inmutables, por lo que no puedes modificar sus elementos una vez creadas.

- Puedes concatenar tuplas con + o replicarlas con \*: tupla + (6, 7), tupla \* 2.

**5. Métodos comunes:**

- count(): Cuenta cuántas veces aparece un valor.

- index(): Encuentra el índice de un valor específico.

**6. Uso:**

- Ideal para colecciones de elementos que no deberían cambiar, como coordenadas o datos constantes.

**Diccionarios**

**1. Definición:**

- Colección no ordenada, mutable, e indexada por claves.

- Las claves deben ser únicas y de tipo inmutable (como cadenas, números, tuplas).

**2. Creación:**

python

diccionario = {"nombre": "Juan", "edad": 25, "ciudad": "Madrid"}

**3. Acceso a elementos:**

- Por clave: diccionario["nombre"].

- Con método get(): diccionario.get("edad"), útil para evitar errores si la clave no existe.

**4. Modificación:**

- Cambiar el valor de una clave existente: diccionario["edad"] = 26.

- Añadir o actualizar claves: diccionario["profesión"] = "Ingeniero".

- Eliminar claves con del: del diccionario["ciudad"].

**5. Métodos comunes:**

- keys(): Devuelve un iterable de las claves.

- values(): Devuelve un iterable de los valores.

- items(): Devuelve un iterable de pares clave-valor.

- pop(): Elimina una clave y devuelve su valor.

**6. Uso:**

- Ideal para almacenar datos que se acceden por una clave específica, como registros de base de datos o configuraciones.

**Comparación rápida**

**- Listas:**

- Ordenadas.

- Mutables.

- Indexadas por números enteros.

**- Tuplas:**

- Ordenadas.

- Inmutables.

- Indexadas por números enteros.

**- Diccionarios:**

- No ordenados (hasta Python 3.7, donde se mantuvo el orden de inserción).

- Mutables.

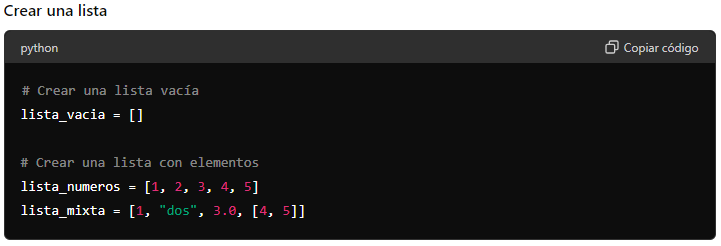
- Indexados por claves.

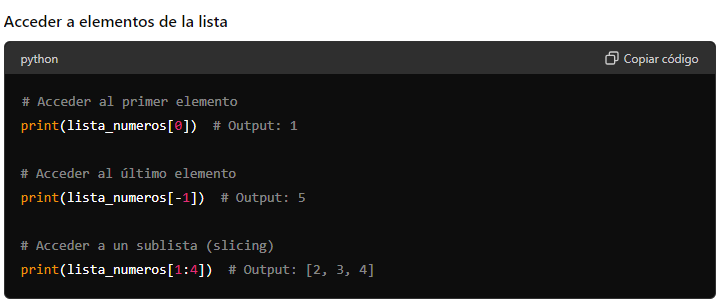
Estos aspectos te ayudarán a elegir la estructura de datos adecuada en función de tus necesidades específicas en Python.

**EJEMPLOS**

**LISTAS**

Las listas son colecciones ordenadas y mutables que permiten duplicados.



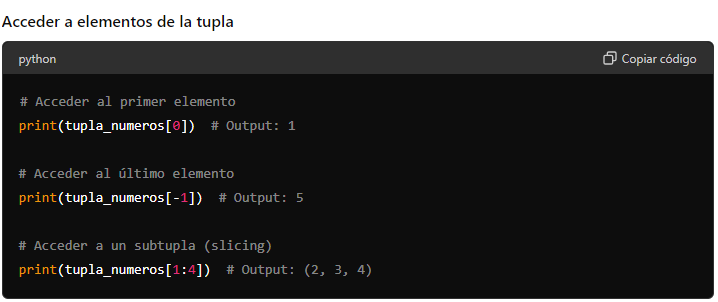




**TUPLAS**

Las tuplas son colecciones ordenadas e inmutables que permiten duplicados.





**DICCIONARIOS**

Los diccionarios son colecciones no ordenadas, mutables e indexadas por claves.

