


Python 3 ▪ Algunas operaciones con estructuras de datos

	LISTA	TUPLA	CONJUNTO	DICCIONARIO
Características	Datos heterogéneos. Acepta repetidos. Elementos mutables y accesibles por índice.	Datos heterogéneos. Acepta repetidos. Elementos inmutables y accesibles por índice.	Datos heterogéneos. Elementos de tipos inmutables, no accesibles por índice. No admite repetidos.	Claves de tipos inmutables, únicas (no admite repetidas). Valores de cualquier tipo, accesibles por clave, admiten repetidos. Pares heterogéneos.
Crear	<code>[]</code> ó <code>list()</code>	<code>()</code> ó <code>tuple()</code>	<code>set()</code>	<code>{}</code> ó <code>dict()</code>
Inicializar con datos	<ul style="list-style-type: none"> <code>[elemento1, elemento2, elemento3]</code> <code>list(iterable)</code> para crear con los elementos de una secuencia o contenedor. 	<ul style="list-style-type: none"> <code>elemento1, elemento2, elemento3</code> (los paréntesis son opcionales) <code>tuple(iterable)</code> para crear con los elementos de una secuencia o contenedor. 	<ul style="list-style-type: none"> <code>{elemento1, elemento2, elemento3}</code> <code>set(iterable)</code> para crear con los elementos de una secuencia o contenedor. 	<ul style="list-style-type: none"> <code>{clave1: valor1, clave2: valor2, clave3: valor3}</code> <code>dict([(c1,v1), (c2,v2)])</code>
Insertar elementos	<ul style="list-style-type: none"> <code>lista.append(elemento)</code> agrega elemento al final <code>lista.insert(posición, elemento)</code> inserta elemento en posición del índice. 	Sólo es posible inicializarla en la creación. Luego no se puede agregar, porque es inmutable.	<ul style="list-style-type: none"> <code>conjunto.add(elemento)</code> donde elemento es un solo dato. 	<ul style="list-style-type: none"> <code>dicc[clave]=valor</code> <code>dicc.update({c3:v3, c4:v4})</code>
Acceder a un elemento	<ul style="list-style-type: none"> <code>lista[índice]</code> lectura/escritura. <code>lista[inicio:fin:paso]</code> Iterando por sus elementos. 	<ul style="list-style-type: none"> <code>tupla[índice]</code> sólo lectura <code>tupla[inicio:fin:paso]</code> Iterando por sus elementos. 	Iterando por sus elementos (no soporta índices). No es posible modificar elementos.	<ul style="list-style-type: none"> <code>dicc[clave]</code> <code>dicc.get(clave, val_defecto)</code> Iterando por sus elementos.
Iterar usando el índice	<code>for i in range(len(lista)):</code> <code>print(lista[i])</code>	<code>for i in range(len(tupla)):</code> <code>print(tupla[i])</code>	No tiene índice.	No tiene índice (el "índice" son las claves).
Iterar con for para iterables	<code>for elemento in lista</code>	<code>for elemento in tupla</code>	<code>for elemento in conjunto</code>	<ul style="list-style-type: none"> <code>for clave in dicc.keys()</code> <code>for valor in dicc.values()</code> <code>for c,v in dicc.items()</code>
Pertenencia	<code>elemento in lista</code>	<code>elemento in tupla</code>	<code>elemento in conjunto</code>	<ul style="list-style-type: none"> <code>clave in dicc.keys()</code> <code>value in dicc.values()</code>
Eliminar elemento	<ul style="list-style-type: none"> <code>del lista[índice]</code> <code>lista.remove(elemento)</code> elimina la primera ocurrencia 	No es posible porque son inmutables.	<ul style="list-style-type: none"> <code>conjunto.remove(elemento)</code> <code>conjunto.discard(elemento)</code> 	<code>del dicc[clave]</code>
Cantidad de elementos	<code>len(lista)</code>	<code>len(tupla)</code>	<code>len(conjunto)</code>	<code>len(dicc)</code>
Vaciar	<code>lista.clear()</code>	No es posible porque son inmutables.	<code>conjunto.clear()</code>	<code>dicc.clear()</code>