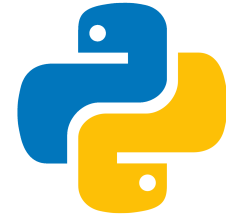


Python [Lists]



1. Python Lists
2. Accedir a valors
3. Actualizar valors
4. Inserir valors
5. Eliminar valors
6. Iterar valors
7. Ordenar llistes
8. Copiar llistes
9. Concatenar llistes
10. Lists methods

Python [Lists]

1. Python [Lists]



1. Python [Lists]

En Python té integrades **quatre tipus de col·leccions** de dades per a **emmagatzemar múltiples valors/objectes en una sola variable iterativa**: Llistes, Tuples, Sets i Dictionaris.

Les llistes **es comporten com un vector** dinàmic tal i com podem trobar en altres llenguatges.

1. Python [Lists]



1. Python [Lists]

- Les llistes es representen amb claudators: `cars = ["Audi", "BMW", "Volkswagen"]`
- Les llistes son **indexades**.
- El primer element té **índex 0** i consecutivament.
- Les llistes son ordenades ja que accedim als valors usant un índex; però els seus valors es guardaran per ordre d'inserció.
- Els **valors** de les llistes **poden ser actualitzats**.
- Les llistes accepten **insercions, eliminacions i modificacions** dels seus ítems.
- Les llistes **accepten valors repetits**.
- Una lista **pot contenir elements de diferent tipatge**: `llista = ["abc", 34, True, 40, "male"]`

Accedir a valors

2. Python [Lists]: accedir a valors



2. Accedir a valors

- Donada una llista: `cars = ["Audi", "BMW", "Volkswagen"]`

Accedir	Python code
Un valor usant índex (<i>error si 'out of bounds'</i>)	<code>print(cars[2])</code>
Un valor usant índex negatiu (<i>últim element</i>)	<code>print(cars[-1])</code>
Múltiples valors usant rang (<i>tres opcions</i>) <i>*(inclou el primer índex del rang pero no l'últim!)</i> <i>** (si obviem el final del rang, mostra fins al final)</i>	<code>print(cars[0:2]) /</code> <code>print(cars[:2]) /</code> <code>print(cars[1:])</code>
Buscar un valor dins de la llista	<code>if "BMW" in cars:</code> <code> print("Sí, hi ha un BMW!")</code>

Actualizar valores

3. Python [Lists]: actualitzar valors



3. Actualitzar valors

- Donada una llista: `cars = ["Audi", "BMW", "Volkswagen"]`

Actualitzar	Python code
Un valor usant un índex	<code>cars[1] = "Renault"</code>
Un rang d'elements Canviarà els elements cars[0] i cars[1] <i>*(Recorda que l'índex final del rang no està inclòs)</i>	<code>cars[0:2] = ["Renault", "Tesla"]</code>
Canviar un element per dos <i>*Canviem el valor "Audi" per "Citroën" i "Tesla" fent créixer la llista amb un element més. Es pot fer també l'acció inversa: canviar dos elements per uno.</i>	<code>cars[0:1] = ["Citroën", "Tesla"]</code>

Insérer valeurs

4. Python [Lists]: inserir valors



4. Inserir valors

- Donada una llista: `cars = ["Audi", "BMW", "Volkswagen"]`

Inserir	Python code
Element(s) al final	<code>cars.append("Dacia")</code>
Element(s) a un índex	<code>cars.insert(2, "Peugeot")</code>
Una llista (o una altra estructura iterativa) al final	<code>cars.extend(["Lamborghini", "Lexus"])</code>

Eliminar valores

5. Python [Lists]: eliminar valors



5. Eliminar valors

- Donada una llista: `cars = ["Audi", "BMW", "Volkswagen"]`

Eliminar	Python code
Un valor	<code>cars.remove("Audi")</code>
Un element per índex (<i>dues opcions</i>)	<code>cars.pop(2) / del cars[2]</code>
L' últim element	<code>cars.pop()</code>
Tots els valors	<code>cars.clear()</code>
Tots els valors i l'estructura de dades	<code>del cars</code>

Iterar valores

6. Python [Lists]: iterar valores



6. Iterar valores

Iterar	Python code
while	<pre>while i < len(cars): print(cars[i]) i = i + 1</pre>
for	<pre>for i in range(len(cars)): print(cars[i])</pre>
for-each	<pre>for c in cars: print(c)</pre>
for-enumerate	<pre>for i, c in enumerate(cars): print(i, "->", c)</pre>
<u>List Comprehensive</u> shorthand	<pre>[print(c) for c in cars]</pre>

<https://docs.python.org/2/tutorial/datastructures.html#looping-techniques>

Ordenar valores

7. Python [Lists]: ordenar valors



7. Ordenar valors

- Donada una llista: `cars = ["Audi", "BMW", "Volkswagen"]`

Ordenar	Python code
Numèrica o alfabèticament	<code>cars.sort()</code>
Descendentment (<i>dues opcions</i>)	<code>cars.sort(reverse = true)</code> / <code>cars.reverse()</code>
Alfabèticamente case insensitive	<code>cars.sort(key = str.lower)</code>

Copiar llistes



8. Copiar llistes

- Donada una llista: **`cars = ["Audi", "BMW", "Volkswagen"]`**
- A Python una llista no es pot copiar usant l'operador d'assignació '='. Això provoca una copia per referència ja que dues variables apuntaran a la mateixa estructura de dades guardada en RAM, però no una copia de valors.
- Per crear una nova variable que contingui la mateixa estructura i valors que una llista donada, tenim dues opcions:

`carsB = cars.copy()`

`carsC = list(cars)`

Concatenar llistes



9. Concatenar llistes

- Donades dues llistes: `cars = ["Audi", "BMW", "Volkswagen"]`
`motos = ["Vespa", "Honda", "Suzuki"]`

- Disposem de tres maneres de concatenar-les

`vehícules = cars + motos` `''' Crea una nova llista amb els valors de cars i motos '''`

`cars.extend(motos)` `''' Afegeix els valors de motos al final de cars '''`

`for c in motos:` `''' Afegeix els valors de motos al final de cars '''`
 `cars.append(c)`

[Lists] methods

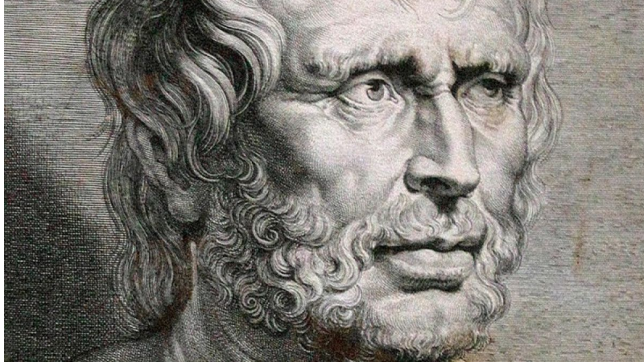
10. Python [Lists]: methods



10. [Lists] methods

Method	Description
<u>append()</u>	Adds an element at the end of the list
<u>clear()</u>	Removes all the elements from the list
<u>copy()</u>	Returns a copy of the list
<u>count()</u>	Returns the number of elements with the specified value
<u>extend()</u>	Add the elements of a list (or any iterable), to the end of the current list
<u>index()</u>	Returns the index of the first element with the specified value
<u>insert()</u>	Adds an element at the specified position
<u>pop()</u>	Removes the element at the specified position
<u>remove()</u>	Removes the item with the specified value
<u>reverse()</u>	Reverses the order of the list
<u>sort()</u>	Sorts the list

Font: https://www.w3schools.com/python/python_lists_methods.asp



“Mediocre és aquell que exigeix molt als altres i poc a sí mateix.”

Séneca (4 aC – 65 dC)

