List.append(e): añade el elemento e a la lista “list”.

*------------------------------------------------------------------------------------------------------------------------------*

List. Extend (l2): añade a la lista “list” la lista “l2”, la modifica. No es lo mismo que list + l2, que únicamente nos devuelve el resultado pero no se guarda en list, list seguirá siendo igual, no se modifica.

*------------------------------------------------------------------------------------------------------------------------------*

List.insert(posición, e), inserta el valor e en la posición indicada.

*------------------------------------------------------------------------------------------------------------------------------*

List.remove(e): borra el primer valor e que encuentre.

*------------------------------------------------------------------------------------------------------------------------------*

List.pop(): extrae el último valor si no pones nada entre paréntesis, si pones una posición (p.e. [1]) entre paréntesis, extraerá esa posición [1].

*------------------------------------------------------------------------------------------------------------------------------*

List.count() : únicamente encuentra elementos iguales, por ejemplo, si en una lista tenemos Aragón, y hacemos list.count(A), no va a encontrar A. Además, si el elemento está dentro de otra cadena, no lo encontrará. P.e: l3 = [1, [2, 3], 4, [5, 6], 7, 8, 9] list.count(2) devuelve 0.

*------------------------------------------------------------------------------------------------------------------------------*

List.index(e): nos devuelve la posición en la que está el elemento e. Nos tenemos que asegurar que existe, de lo contrario, dará un error.

*------------------------------------------------------------------------------------------------------------------------------*

list.index(e) if e in list else -1: “Si e está en list, devuélveme dónde está. De lo contrario, devuelve -1”.

*------------------------------------------------------------------------------------------------------------------------------*

3 in l3: devuelve un booleano

*------------------------------------------------------------------------------------------------------------------------------*

List.reverse(): da la vuelta a la lista, NO ordena. Es equivalente a [::-1]

*-----------------------------------------------------------------------------------------*

List.sort() ordena la lista. Para ordenarla en orden descendente, sería list.sort(reverse=True)

*------------------------------------------------------------------------------------------------------------------------------*

**Tanto reverse como sort, no se almacenan en ninguna variable. Es una orden que modifica la lista que tenemos, pero que no devuelve nada.**

*------------------------------------------------------------------------------------------------------------------------------*

**COPIAS DE LISTAS:**

Old\_list = new\_list (Copia contectada, cuando se modifique una, se modificara la otra)

Old\_list = new\_list[:] (Copia no contectada, si modificamos new\_list NO se modificará old\_list. Es quivalente a old\_list=new\_list.copy o old\_list=list(new\_list.copy). [:] significa que los valores que se cogen de esa lista son todos, realmente es un rango, que al no especificar valores de inicio y fin, coge todos.

**Tuplas**

*T = (True, False, True)*

*------------------------------------------------------------------------------------------------------------------------------*

*Any(t) #¿Algún true?*

*True*

*------------------------------------------------------------------------------------------------------------------------------*

*All(t) ¿Son todas true?*

*False*

**LAS TUPLAS SON INVARIABLES, NO SE PUEDEN MODIFICAR. Sin embargo, sí que se puede cambiar el valor de una lista.**

*T = (1, 2, [1, 8, 16], 4, 5, 6)*

*T[2] = 0 -🡪* ***error***

*T[2][0] = 0 🡪 ahora t= (1, 2, [0, 8, 16], 4, 5, 6)*

**EL PRIMER NIVEL ES EL QUE IMPORTA**

**RANGOS Y RANGE()**

\*range(4) 🡪Saca sus elementos.

*[\*range(4), 4, 5, 6]*

*[0, 1, 2, 3, 4, 5, 6]*

**PILAS Y COLAS**

PILAS 🡪 LIFO (Last In, First Out): El último en llegar, es el primero en salir.

COLAS 🡪 FIFO (First In, First Out): Orden más intuitivo, el primero en llegar, será el primero en salir.

**TRANSFORMACIONES ÚTILES**

Str.replace() 🡪 Para reemplazar algo de una cadena por otra cosa. No es una orden, no se guarda automáticamente en la variable original. Pista: El intérprete nos devuelve algo, si no nos hubiese devuelto nada, hubiera sido una orden.

*s.replace("hola","hello")*

*Hello, mundo*

Str.split()🡪 s=’hola “mundo”’’ (No tiene en cuenta espacios, ni tabulaciones, ni saltos de línea)

*s.split()*

*[‘hola’,’”mundo”’]*

Str.strip()🡪 elimina espacios del principio y del final.

Str.join()🡪 Te devuelve la unión con la separación indicada

"".join(tuple('a', 'b', 'c')) # devuelve 'abc'

"\_\_\_\_\_".join(tuple('a', 'b', 'c')) # devuelve 'a\_\_\_\_b\_\_\_\_c'

Str.capitalize()🡪Transforma únicamente el primer carácter en mayúsculas. Sí que tiene en cuenta si hay espacios, por lo que habría que hacer s.strip().capitalize()

Str.casefold()🡪 Pasa a minúscula incluso caracteres equivalentes, como la beta en alemán, que son ss.

**DICCIONARIOS**

No tienen slicing porque no tiene sentido, se accede a los elementos de uno en uno. Son mutables, es decir, se pueden modificar al contrario de las tuplas. Para crear elementos en el diccionario, en vez de utilizar la función append, lo que se hace es utilizar la misma sintaxis que para modificar la información, haciendo referencia a una nueva posición z:

*dic [‘z’]=5*

*Crear diccionarios*

*---------------------------------------------------------------------------------------------------------------------------*

*dic3 = { 'nombre': ‘Pachi’, ‘apellido': Sucunza }*

*todos = {‘Profesor’:dic3, ‘Alumno favorito’: {‘Nombre’: ‘Pepito’, ‘Apellido’: ‘Pérez’}}*

*todos [‘Alumno favorito’]*

*Te devuelve {‘Nombre’: ‘Pepito’, ‘Apellido’: ‘Pérez’}*

*Todos[‘Profesor’][‘Apellido’] te devuelve ‘Sucunza’*

**CONJUNTOS**

Versión reducida de diccionarios en la que únicamente tienes claves (keys).