**¿Cuál es la diferencia entre una lista y una tupla en Python?**

La diferencia entre estos dos conceptos es la mutabilidad.

Las listas son mutables, es decir, podemos modificarlas añadiendo, eliminando o incluso cambiando el contenido de estas.

Sin embargo, las tuplas son inmutables, una vez creadas no podremos modificarlas.

Para diferenciarlas a simple vista nos fijaremos si están entre corchetes (estas serán las listas) o entre paréntesis (estas serán las tuplas).

Por lo tanto, dependiendo la utilidad que les vayamos a dar, nos convendrá usar una u otra.

Si tenemos claro que no vamos a modificar nada, podremos usar la tupla. Y si, por el contrario, desde un principio sabemos que vamos a necesitar modificar la información, usaremos la lista.

Ejemplo:

Lista = [1, 2, 3]

Tupla = (a, b, c)

**¿Cuál es el orden de las operaciones?**

Para responder a esta pregunta empezaré por mencionar las siguientes siglas: PEMDAS/PEDMAS. Para poder recordar esta nemotecnia, la completaremos con la siguiente frase: Please Excuse My Dear Aunt Sally.

Con esto lo único que quiero mostrar es cual es el orden de las operaciones, a continuación, muestro su equivalencia:

Please -> Parans ()

Excuse -> Exponents **

My -> Multiplication *

Dear -> Division /

Aunt -> Addition +

Sally -> Subtraction -

Es muy importante que sigamos este orden porque si no, el resultado será incorrecto. Lo único que no es relevante es el orden de la multiplicación y división.

**¿Qué es un diccionario Python?**

Gracias a los diccionarios podemos crear una clave con su valor correspondiente y almacenarlas en una variable, son muy útiles cuando queremos relacionar datos. Un ejemplo sería el siguiente:

```
profesores = {
  "matemáticas": "Jon",
  "lengua": "Maria",
}
```

Nuestro diccionario esta almacenado dentro de la variable “profesores” y tiene un elemento con la clave “matemáticas” y el valor “Jon”.

A diferencia de las listas, en las que trabajamos con índices, la estructura que utilizamos en los diccionarios es la de clave-valor, es decir, pasamos la clave para acceder al valor.

**¿Cuál es la diferencia entre el método ordenado y la función de ordenación?**

La principal diferencia reside en el almacenamiento de las variables. Ambas opciones sirven para ordenar listas pero, el método ordenado nos permite almacenar el valor dentro de una variable diferente y la función de ordenación no.

En cuanto a la función de ordenación es la siguiente:

```
Sorted()
```

Y el método ordenado lo podemos identificar asi:

```
List.sort()
```

En la función de ordenación, no alteramos los valores, sin embargo, en el método ordenado modificamos la lista inicial.

Por lo tanto, cuando no nos importe modificar la lista original, utilizaremos el método ordenado y, por el contrario, cuando necesitemos que la lista se quede intacta, utilizaremos la función de ordenación.

**¿Qué es un operador de reasignación?**

Utilizamos operadores de reasignación para hacer más simple el código. Como ejemplo podemos poner el siguiente:

Ventas_totales = 200

Si a esto le queremos sumar 50, podríamos hacer:

Ventas_totales = Ventas_totales + 50

Pero si usáramos un operador de reasignación, podríamos simplificarlo de la siguiente manera:

Ventas_totales += 50

De esta forma conseguimos el mismo resultado y el código en menos repetitivo.

Podemos usarlo no solo con sumas, sino que también con restas, multiplicación, división, etc.