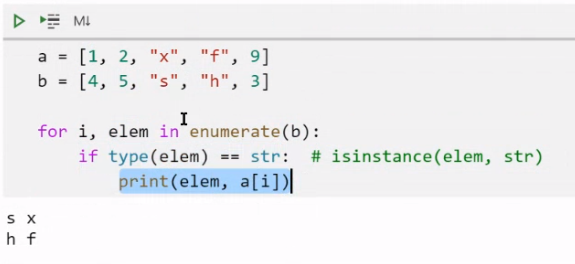
**For con range**

Trabajamos con el rango que queremos de la lista (valor intermedio, de la mitad palante, cada 2 valores etc….)

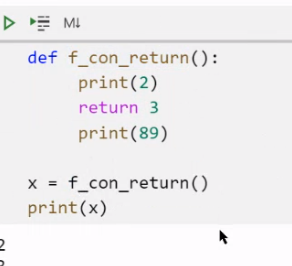
**For con enumerate**

Trabajamos también con su posición. Puede ser el caso de trabajar con 2 listas.

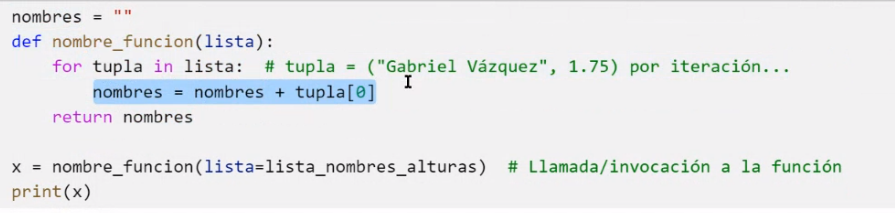


**Return**

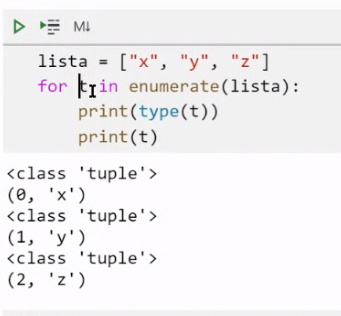
Return siempre nos va a mostrar lo que retorna la función. No el resultado de lo que hay dentro, sino es un valor aplicado a la función que se asigna con return. En este caso, x=3



**TRABAJAR CON VARIABLES GLOBALES EN FUNCION**. Si trabajo con una variable global dentro de una función, no me dejará modificarla con su propio término como se muestra en pantalla. Me dará error. Tengo que utilizar una variable local.



**Enumerate con 1 variable**: En este caso, la variable será valor y posición.



**Diccionario**

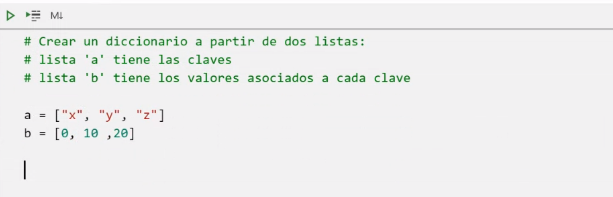
En un diccionario no podemos acceder por posición, para acceder al valor de una clave, tenemos que poner entre corchetes la clave, y nos devolverá el valor [].





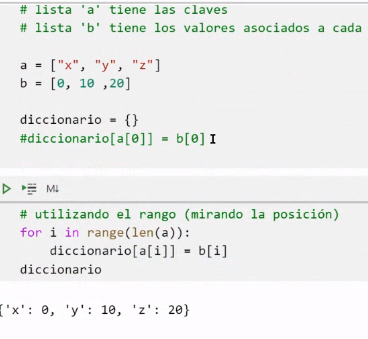
* .items nos coje valores y claves.
* .keys nos coje claves.
* .value nos coje valores.

**Ejercicio**

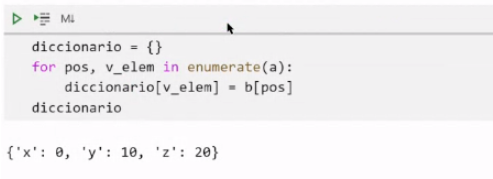


-Primero creamos un diccionario vacío.

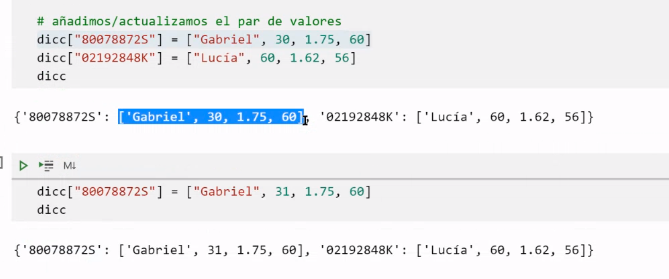
-Después, con un for en rango de longitudes accedemos a cada posición y la asignamos con la posición de b (que como es diccionario, podemos explicar el valor de un elemento si operamos con igualdad).



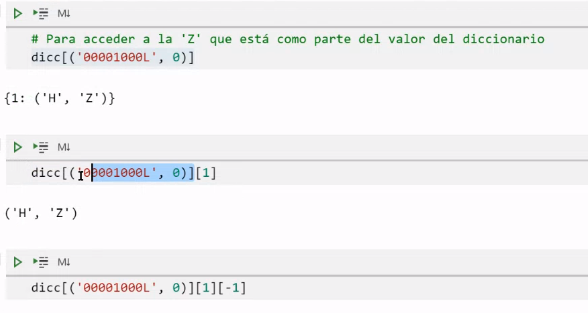
-O utilizando el enumerate



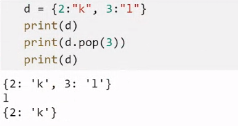
-En el siguiente ejercicio enseñamos el propósito de los diccionarios. Tener alores asignados a una clave individual. Y poder actualizar así sus valores.



-Un tipo lista no puede ser clave, pero sí puede ser su valor. Un diccionario puede ser un string, tupla, un int o un float. Después, podemos asignar a la key del diccionario, un valor con diccionario. Entonces tendremos un diccionario dentro de un diccionario, y tendremos que acceder a él como se muestra a continuación: (suponiendo que tenemos una clave tupla y un valor lista que tiene dentro un diccionario).



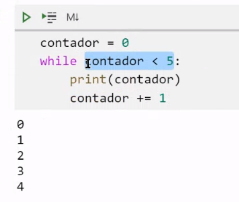
-La función pop me quita una clave, con su valor, en diccionarios.



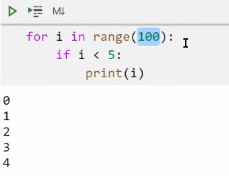
**While**

While es una condición que mientras se mantenga TRUE, se hará siempre. Recordemos, que siempre se puede romper con un break.

Esta es la idea de while:

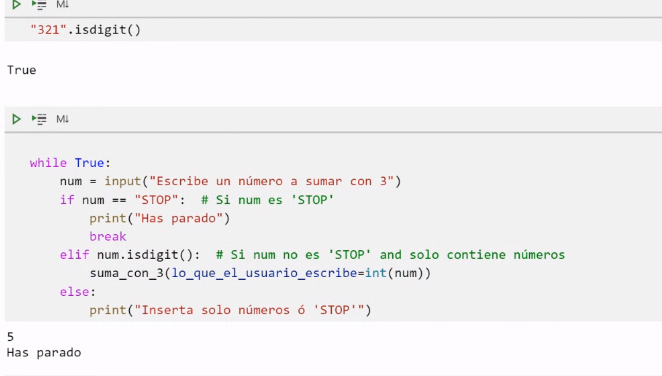


Su versión con for +if:

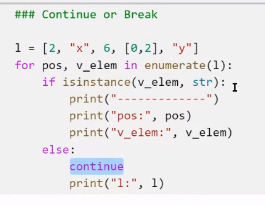


-La primera de las 2 ocipnes de arriba es más eficiente porque el while ha ejecutado 5 veces el comando, mientras q el for ejecutará todas las veces su comando, y carga más la interfaz.

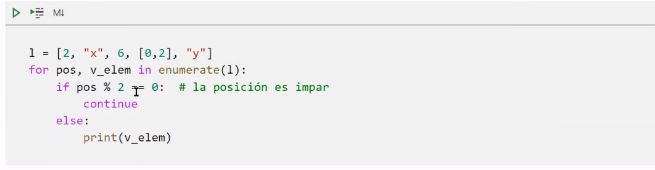
-En este ejercicio, comprobamos con un while si un parámetro solo tiene números (no valen floats).



Continue: Salta a la siguiente iteración.// Y **pass** es sencillamente nada, es dejar un blanco, un espacio sin más. Este puede valer para marcar un if vacío.



-Fijemonos, en este ejercicio, como podemos solucionar el problema que nos daba el while para sacar solo los valores pares o impares con módulos. (Suponiendo que hiciésemos contador) En este ejercicio, si el valor es par, hace continue y salta al for con el siguiente, que sería una posición impar… WAW



-Ejercicio medio haciendose. En este caso, solo quiero que me reconozca números en los strings, utilizo la función isdigit, y además, pongo un if, que me dice que si ve un punto, también lo incluya y lo valide.