Preguntas teoricas del checkpoint 4.

- **¿Cuál es la diferencia entre una lista y una tupla en Python?**
  Aunque los dos elementos tengan similaridades siendo elementos que coleccionan datos estos dos tienen unas cuantas diferencias.
  La primera que salta a la vista es el hecho de que las listas contienen sus datos entre [] mientras que las tuplas usan ().
  Y la segunda de la cual parten muchas otras diferencias es el hecho de que las listas son modificables, mientras que las tuplas son estaticas; es decir, que se puede cambiar, añadir o quitar elementos de una lista, pero si se intenta hacer esto te saldra error. Esto implica que las listas se procesen (relativamente hablando) más despacio debido a los posibles modificaciones y tambien influye en los usos que puedan tener. En el caso de las tuplas su naturaleza inmutable las hace ideales para operaciones rapidas en las que se quiera asegurar que todos los elementos internos esten intactos, como seria el caso con un diccionario.

- **¿Cuál es el orden de las operaciones?**
  Es el orden en el que se hace una calculacion matematica. El orden estandar es habitualmente enseñado usando las siglas PEMDAS (en español seria PEMDSR) para facilitar la memorización del orden.

1. Parentesis ()
2. Exponentes **
3. Multiplicaciones *
4. Divisiones /
5. Sumas +
6. Restas -

Saber el orden correcto es necesario para evitar hacer malas calculaciones, y aún asi, hay debates entre matematicos a día de hoy.

- **¿Qué es un diccionario Python?**
  Un diccionario Python es una colección editable de información en formato indice mediante el uso de una llave (key) y un valor (value). Estos no suelen contar con un orden particular.
  Usando la llave puedes acceder a un valor e incluso hacer cambios a este.

```
the_dic = {'Cell':'Potato', 'Forma':'Cheese', 'Exilus':'Tomato', 'Kuva':'Pepper'}
```

- **¿Cuál es la diferencia entre el método ordenado y la función de ordenación?**
  Las dos cumplen un mismo objetivo pero con una gran diferencia. Las dos operaciones ordenan los items de una lista. Pero, el metodo de ordenado crea una copia de la lista que ha sido ordenada, asi no cambia el original a diferencia de la función de ordenacion la cual modifica la lista permanentemente.

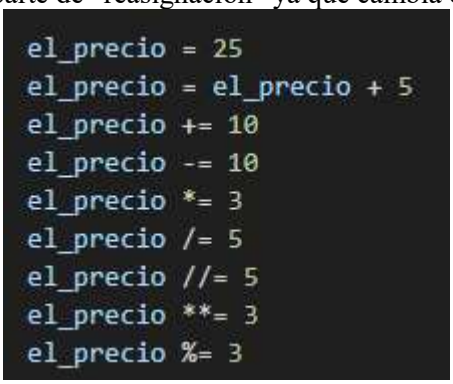
```
lista_ordenado = [99, 27, 13, 48, 3]
lista_chula = sorted(lista_ordenado)
print(lista_chula)
print(lista_ordenado)

lista_ordenacion = ['Metatron', 'Sans', 'Asriel', 'Frisk', 'Chara']
lista_ordenacion.sort()
print(lista_ordenacion)
```

```
[3, 13, 27, 48, 99]
[99, 27, 13, 48, 3]
['Asriel', 'Chara', 'Frisk', 'Metatron', 'Sans']

[Done] exited with code=0 in 0.135 seconds
```

- **¿Qué es un operador de reasignación?**
  Son operadores que permiten ejecutar una operación matematica y guardar el numero resultante en la variable de origen, de ahi la parte de “reasignacion” ya que cambia el valor original de la variable.



```
el_precio = 25
el_precio = el_precio + 5
el_precio += 10
el_precio -= 10
el_precio *= 3
el_precio /= 5
el_precio //= 5
el_precio **= 3
el_precio %= 3
```

Ejemplo de sintaxis de un operador de reasignacion