M2 Checkpoint 4

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

   La mayor diferencia entre una **lista** y una **tupla** es que las **listas son mutables** mientras que las **tuplas son inmutables**. Por ejemplo, si queremos cambiar un elemento, en una lista se nos permitirá, cuando en una tupla no se podrá hacerlo, nos dará error. Si vamos a trabajar con objetos que no cambiarán a lo largo del programa, es recomendable usar las tuplas, si no, usamos listas, donde podremos quitar elementos, cambiar, añadir.

```python
my_list = [1, 2, 3,4]
my_tupla = (1, 2, 3,4)

my_list[1] = 5
print(my_list)
# Output: [1, 5, 3, 4]

my_tupla[1] = 5
print(my_tupla)
# Output: TypeError: 'tuple' object does not support item assignment
```

2. ¿Cuál es el orden de las operaciones?

   El orden de las operaciones en Python se identifica con las siglas **PEMDAS**:

   P - paréntesis
   E - exponente
   M - multiplicación
   D - división
   A - adición o suma
   S - sustracción o resta

3. ¿Qué es un diccionario Python?

   Un **diccionario** en **Python** es una colección de datos que guarda diferentes tipos de datos como **String**, **números enteros**, **decimales**, **listas** o incluso **otros diccionarios**. Los datos se almacenan en pares de **clave** y **valor**, cada clave es única y se usa para acceder a su valor correspondiente.

```python
my_dictionary = {
    'nombre': 'Alison',
    'edad': 30,
    'ciudad': 'Vitoria-Gasteiz'
}

first_key = next(iter(my_dictionary))  # Output: nombre
first_valor = my_dictionary[first_key]  # Output: Alison
```

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

   Tanto el **método sort()** como la **función sorted()** se usan para ordenar listas, pero hay algunas diferencias como por ejemplo:

   El **método sort()** modifica la lista original ordenándola, no devuelve otro nuevo objeto y solo se usa con listas.

```python
my_list = [105, 56, 89, 46, 9, 74]
my_list.sort()
print(my_list)  # Output: [9, 46, 56, 74, 89, 105
```

   La **función sorted()** no modifica la lista original, sino que crea una nueva lista ordenada y funciona con cualquier objeto iterable como listas, tuplas, diccionarios.

```python
my_list = [105, 56, 89, 46, 9, 74]
new_list = sorted(my_list)  # Crea una nueva lista ordenada
print(new_list)  # Output: [9, 46, 56, 74, 89, 105
print(my_list)  # Output: [105, 56, 89, 46, 9, 74]
```

5. ¿Qué es un operador de reasignación?

   Un **operador de reasignación** es un operador que modifica el valor de una variable. El **operador de asignación** más **básico** es **=** y se usa para asignar un valor a una variable. También hay **operadores de reasignación combinados**, como **+=**, **-=**, ***=** etc.

   El **operador de reasignacion** cuando se usa en una **tupla**, aunque la tupla no cambia, se crea una nueva tupla y se reasigna a la tupla inicial, como en el siguiente ejemplo:

```
my_tuple = [10, 11, 12]
my_tuple += [13, 14]  # Crea una nueva tupla y la reasigna a my_tuple
print(my_tuple)  # Output: [10, 11, 12, 13, 14]
```