M2C4 Python Assignment

Preguntas teóricas:

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

En cuanto **a la sintaxis,** para crear una tupla se utilizan paréntesis ( ) mientras que para crear una lista se utilizan corchetes [ ].

TUPLA frutas = **(**'plátano', 'manzana', 'mandarina'**)**	LISTA verduras = **[**'acelga', 'brócoli', 'puerro'**]**

Tanto la lista como la tupla son colecciones en Python, pero **la lista es mutable y la tupla es inmutable.** Por ello, podemos añadir elementos a la lista pero para añadir elementos a la tupla, debemos crear una nueva tupla.

TUPLA frutas = ('plátano', 'manzana', 'mandarina') frutas += ('kiwi',) print(frutas) Output: (‘plátano’, ‘manzana’, ‘mandarina’,’kiwi’)	LISTA verduras = ['acelga', 'brócoli', 'puerro'] verduras.append ('cardo') print(verduras) Output: ['acelga', 'brócoli', 'puerro',’cardo’] ❗ verduras+= ‘cardo’ print(verduras) Output: ['acelga', 'brócoli', 'puerro',’cardo’,’c’, ‘a’,’r’,’d’,’o’]

Tanto en la lista como en la tupla, **los elementos están indexados** y se puede acceder a ellos usando su posición, comenzando desde [0].

TUPLA primera= frutas[0] print(primera) Output: plátano	LISTA primera= verduras[0] print(primera) Output: plátano

Diferencias a la hora de **ordenar (sort) listas y tuplas**:

Las tuplas no responden al método sort(), ya que, como se ha mencionado anteriormente, son inmutables.
En cambio, las listas sí pueden ordenarse alfabéticamente o de forma numérica (de menor a mayor o de mayor a menor) utilizando el método sort().

Las tuplas ocupan menos memoria que las listas, lo que las hace más eficientes y rápidas en ciertos contextos. Sin embargo, son inmutables, es decir, no se pueden modificar una vez creadas.

Por eso, se utilizan listas cuando se necesita añadir, eliminar o modificar elementos, y tuplas cuando se quieren proteger los datos para que no cambien, como en el caso de coordenadas, claves constantes o configuraciones fijas.

¿Cuál es el orden de las operaciones?
El orden de las operaciones es el utilizado en la aritmética, algo básico en matemáticas.
En el curso se utiliza un método para poder recordar cual es el orden que debemos tener en cuenta a la hora de realizar operaciones. PEMDAS

Please➡️PARENTHESIS () (PARENTESIS)
Excuse➡️EXPONENTS ** (EXPONENTE)
My ➡️MULTIPLICATION* (MULTIPLICACION)
Dear ➡️DIVISION /   (DIVISION)
Aunt ➡️ ADDITION +   (SUMA)
Sally➡️SUBSTRACTION - (RESTA)

De esta manera, en una operación se da prioridad a realizar primero las operaciones dentro de los paréntesis. Dentro del paréntesis tendrán prioridad las operaciones en el orden mencionado. Con los resultados de las operaciones en paréntesis, se seguirá con la operación teniendo en cuenta el orden.

Ejemplo:

```
resultado = 5 + 2 * (3 ** 2)
print(resultado)

Output: 23
```

Primero se resuelve el paréntesis: **3∗∗2=9**
Luego la multiplicación: **2∗9=18**
Finalmente la suma: **5+18=23**
Resultado: **23**

¿Qué es un diccionario Python?

Un diccionario en Python es una colección de datos no ordenada, mutable y con claves (key) y valores (value) en la que cada clave está asociada a un valor como si fuesen entradas de palabras y sus acepciones o definiciones. En los pares de clave y valor, las claves deben ser únicas.

En cuanto a la sintaxis, para crear un diccionario se utilizan llaves { }, la clave(key) va entre doble comilla y después de dos puntos : va el valor(value) que puede ser un valor único o una colección tupla o lista.

```
grupos = {
    "uno": [ "Ana",  "Juan",  "Isabel"],
    "dos": [ "Nerea",  "Jon",  "Bea"],
   "tres": [ "Aitor",  "Mikel",  "Eli"]
}


print(grupos)
```

Puedes acceder a un valor del diccionario mediante su clave:

```
print(grupos['uno'])
```

En los diccionarios se pueden añadir nuevos pares de clave y valor

```
grupos["cuatro"] = ["Laura", "Pedro"]
```

Se pueden modificar los valores de una clave existente.

```
grupos["uno"].append("Carlos")
```

O borrar pares de clave y valores

```
del grupos[‘tres’]
```

¿Cuál es la diferencia entre el método ordenado y la función de ordenación? Sort vs Sorted

Utilizando el método de ordenación SORT, la lista original se modifica.

```
exercise = ['uno', 'dos', 'tres', 'cuatro']
exercise.sort()
print(exercise)
```

Utilizando SORTED, mantenemos la lista original creando una nueva ordenada.

```
exercise = ['uno', 'dos', 'tres', 'cuatro']
sorted_list = sorted(exercise)
print(sorted_list)
```

¿Qué es un operador de reasignación? (assignment operators)
Los operadores en Python pueden utilizarse de dos maneras: la primera es utilizando los operadores tal y como se utilizan en matemáticas y la segunda sería utilizando operadores de reasignación (variable [operador]=[número]).

(variable=variable[operador][número]) es igual a (variable [operador]=[número])

OPERACIONES	OPERADORES DE ASIGNACIÓN
total=total+10 total=total-10 total=total*2 total=total/10 total=total// total=total**2 total=total%2	total += 10 total -= 10 total *= 2 total /= 10 total //= 10 total **= 2 total %= 2