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

List y Tuple en Python son clases de estructura de datos de Python. Una lista es dinámica, mientras que una tupla tiene características estáticas. Esto significa que las listas se pueden modificar mientras que las tuplas no se pueden modificar; una tupla es más rápida que una lista debido a su naturaleza estática. Las listas se indican entre corchetes y las tuplas, entre paréntesis.

Lista	**Tupla**
Las listas son mutables	Las tuplas son inmutables
Las consecuencias de las iteraciones llevan mucho tiempo	Las consecuencias de las iteraciones son comparativamente más rápidas
Una lista es mejor para realizar operaciones como inserción y eliminación	El tipo de datos Tuple es adecuado para el acceso a elementos
Las listas consumen más memoria	Una tupla consume menos memoria en comparación con una lista
Las listas tienen varios métodos integrados	Tuple no tiene muchos métodos integrados
Es más probable que se produzcan cambios inesperados y errores	Esto es difícil de hacer en una tupla

```
# Ok
my_list = [1, 2, 4, 4, 3, 3, 3, 6, 5]
my_list[3] = 77

# Error
my_tuple = (0, 1, 2, 3)
my_tuple[0] = 4
```

2. ¿Cuál es el orden de las operaciones?

Las operaciones de mayor prioridad están en la parte superior, en la parte inferior, con baja prioridad. Los cálculos se realizan de izquierda a derecha, es decir, si una expresión contiene operadores de la misma precedencia, se ejecutará primero el de la izquierda.

El operador de exponenciación es una excepción a esta regla. De los dos operadores **, primero se ejecutará el derecho y luego el izquierdo.

() - Soportes
** - Exponenciación
+x, -x, ~x - Unario más, menos y negación bit a bit
*, /, //, % - Multiplicación, división, división entera, resto de división
+, - - Suma y resta

≪, ≫ - Cambios de bits
& - poco y
^ - OR exclusivo bit a bit (XOR)
| - poco O
==, ≠, >, ≥, <, ≤, - es, no está, en, no en Comparación, verificación de identidad, verificación de ocurrencia
no - lógico NO
and - lógico Y
or - lógico O

3. ¿Qué es un diccionario Python?

Un diccionario es una estructura de datos que le permite registrar y recuperar diferentes valores por clave. En algunos idiomas, las claves y valores de un diccionario sólo pueden ser objetos específicos. Y en Python, cualquier entidad: hasta clases de usuario.

```
# Crear
my_new_dict = dict()
another_dict  = {"string_key": "the value", 2: "another value"}

# Expresión generadora
list_one = ["one", "two", "three", "four", "five"]
list_two = [1, 2, 3, 4, 5]
the_dict = {}
the_dict2 = {}

for k, v in zip(list_one, list_two):
    the_dict[k] = v + 1

the_dict2 = dict(zip(list_one, list_two))

# Obtener valores
let_a = the_dict[0]
let_b = the_dict[1]

# Agregar, cambiar y eliminar elementos
the_dict["my_key"] = "some value"
the_dict["my_another_key"] = 42

the_dict["my_key"] = "yet another value"

del the_dict["my_key"]

# Consultar disponibilidad de llaves
print("some value" in the_dict)

print("some value 2" in the_dict)
```

```
# Copiar diccionario
dict_copy = the_dict.copy()

# Determinar la longitud del diccionario
length = len(the_dict)

# Iteración del diccionario
for k in the_dict.keys():
    print(k)
    print(the_dict[k])

for v in the_dict.values():
    print(v)

# Diccionarios anidados
nested_dict = {"first_color": "blue", "second_color": "red"}
the_dict["color"] = nested_dict
```

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

Ordenar es el proceso de organizar elementos de una colección en un orden específico. En Python, puedes usar el método sort() para listas y la función sorted() para varios iterables para ordenar colecciones. Ambas funciones le permiten ordenar elementos en orden ascendente o descendente, según el parámetro especificado.

La función sorted() devuelve una nueva lista ordenada de la colección iterable que se le pasa. La función devuelve una lista ordenada sin cambiar el orden de los elementos de la colección original.

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
sorted_numbers = sorted(numbers)
```

El método sort() es un método integrado en Python que se utiliza para ordenar los elementos de una lista en su lugar, es decir, cambia la lista original sin crear una nueva lista.

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
numbers.sort()
```

El método sort() y la función sorted() también tienen un parámetro inverso adicional que se puede utilizar para ordenar colecciones en orden inverso.

5. ¿Qué es un operador de reasignación?

Las tuplas son inmutables, lo que significa que no se pueden cambiar, agregar ni eliminar elementos una vez creada la tupla.

Pero existen algunas soluciones.

```
# Convert to list
my_tuple = (“apply”, “banan”, “cherry”)
my_list = list(my_tuple)
my_list[1] = "kiwi"
my_list.append("potato")
my_tuple = tuple(my_list)

# reasignación
my_tuple += (“cheese”,)
```