**1) . What is the difference between enclosing a list comprehension in square brackets and parentheses?**

Square brackets ([]) are used to create a list comprehension, resulting in the creation of a list.

Parentheses (()) are used to create a generator expression, resulting in the creation of a generator object.

Example using list comprehension: [x\*\*2 for x in range(5)]

Example using generator expression: (x\*\*2 for x in range(5))

The list comprehension creates a list with all elements in memory, while the generator expression produces values on-the-fly when iterated.

**2) What is the relationship between generators and iterators?**

Generators are a type of iterator. All generators are iterators, but not all iterators are generators.

Generators use the yield statement to produce a series of values lazily (one at a time) when iterated.

Iterators, in general, are objects that implement the \_\_iter\_\_() and \_\_next\_\_() methods.

**3) What are the signs that a function is a generator function?**

Presence of the yield statement in the function.

Generator functions use yield to produce a series of values, suspending the function's state between calls.

**4) What is the purpose of a yield statement?**

The yield statement is used in generator functions to produce a value and temporarily suspend the function's execution.

It allows the function to yield values one at a time, maintaining the state of local variables between successive calls.

The generator is paused after each yield, and the state is retained, making it memory-efficient for large datasets.

**5) What is the relationship between map calls and list comprehensions? Make a comparison and contrast between the two.**

Map:

The map function applies a given function to all items in an iterable and returns an iterator of the results.

Example: map(lambda x: x\*\*2, range(5))

List Comprehension:

A list comprehension creates a new list by applying an expression to each item in an iterable and optionally filtering the items based on a condition.

Example: [x\*\*2 for x in range(5)]

Comparison and Contrast:

Similarities:

Both map and list comprehensions transform each element in an iterable using an expression or function.

Differences:

map returns an iterator, while a list comprehension returns a list.

List comprehensions offer more expressive power, allowing filtering and nested expressions.

map can be more memory-efficient for large datasets as it produces values on-the-fly.

List comprehensions may be more readable for simple transformations and filtering.