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

Ans - Using square brackets square creates a list comprehension, while using parentheses creates a generator expression. list\_comp is a list containing the elements from 0 to 9, while gen\_exp is a generator object that can produce these elements one at a time when iterated over. Lists store all elements in memory, while generators produce elements on-the-fly, making them memory-efficient for large datasets.Top of Form

1. What is the relationship between generators and iterators?

Ans- Generators are a specific type of iterator in Python. All generators are iterators, but not all iterators are generators. Iterators are objects that implement the iterator protocol, which means they support the \_\_iter\_\_() and \_\_next\_\_() methods. Generators are a convenient way to create iterators using a special syntax involving functions and the yield keyword.

Generators are a type of iterator that allows you to define iterator objects using concise and readable syntax, making them a powerful tool for creating iterators in Python.

1. What are the signs that a function is a generator function?

Ans- A function is a generator function if it contains the yield keyword at least once within its body. When a function uses yield, it becomes a generator function, capable of producing a sequence of values one at a time using the yield statement.

1. What is the purpose of a yield statement?

Ans - The purpose of a yield statement is to temporarily suspend the execution of a function, returning a value to the caller while maintaining the function's state.

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

Ans - Map calls and list comprehensions serve similar purposes, allowing transformation of elements in a sequence. However, list comprehensions offer a more expressive and concise syntax, making them easier to read and write. On the other hand, map calls provide lazy evaluation, potentially saving memory by not creating the entire transformed list immediately. Despite this difference, both approaches are valuable tools in Python for data transformation tasks.