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

**Ans**: In Python, enclosing a list comprehension in square brackets ([...]) results in a list, while enclosing it in parentheses ((...)) creates a generator expression. Lists are fully constructed in memory, consuming more space, while generators produce values on-the-fly, conserving memory.

1. What is the relationship between generators and iterators?

**Ans**: Generators and iterators are closely related concepts in Python. A generator is a specific type of iterator. While iterators are objects with \_\_iter\_\_ and \_\_next\_\_ methods, generators are a concise way to create iterators using functions with the yield statement.

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

**Ans**: A function is identified as a generator function if it contains the yield statement. This statement allows the function to yield values one at a time, maintaining its state between successive calls.

1. What is the purpose of a yield statement?

**Ans**: The yield statement in a generator function serves the purpose of producing a value to the caller while preserving the function's state. It allows the function to be paused and resumed, generating values on demand

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 are both used to apply a function to each element of an iterable. Map returns a map object, and list comprehensions create a list. The primary difference is that list comprehensions produce the entire list at once, while map produces values one at a time. List comprehensions may use more memory, whereas map, combined with list(), may be more memory-efficient for large datasets.