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

Ans: Square brackets: When you use square brackets, you create a list.

Parentheses: When you use parentheses, you create a generator object. This is known as a generator expression. The generator produces values on-the-fly without storing them in memory.

2) What is the relationship between generators and iterators?

Ans: all generators are iterators, but not all iterators are generators. Generators are a specific type of iterator that allows for the dynamic generation of values, making them useful for creating efficient and on-the-fly data processing pipelines, especially for large or infinite sequences.

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

Ans: the key signs that a function is a generator function in Python are:

1. Presence of one or more yield statements within the function.
2. Absence of return statements (although they can be present to indicate the end of iteration).
3. When called, the function returns a generator object that can be iterated over.

4) What is the purpose of a yield statement?

Ans: In Python, the yield statement is used in generator functions to pause the function's execution and yield a value to the caller. When a generator function is called, it returns a generator object that can be iterated over to retrieve values one at a time. The yield statement is crucial for defining the sequence of values that the generator produces.

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

* Ans: Both map calls and list comprehensions serve the purpose of applying a function to each item in an iterable.
* The map function returns an iterator, while list comprehensions directly create a new list.
* map is typically less readable than list comprehensions, especially when using lambda functions.
* List comprehensions are generally faster for simple operations and smaller datasets, while map might be more efficient for large datasets due to lazy evaluation.