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

Ans1

Enclosing a list comprehension in square brackets [] creates a list, while enclosing it in parentheses () creates a generator object.

2) What is the relationship between generators and iterators?

Ans2

An iterator is an object that implements the iterator protocol requires it to have two methods \_\_iter\_\_() and \_\_next\_\_().

Generators are a type of iterator that are defined using a special syntax involving the yield keyword. A generator function is a function that contains one or more yield statements and returns a generator object when called.

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

Ans3

A generator function is a function that contains one or more yield statements and returns a generator object when called. Here are some signs that a function is a generator function

The function contains one or more yield statements.

The function uses the yield keyword instead of return to return values.

The function returns a generator object.

def squares():

for i in range(1, 6):

yield i\*\*2

4) What is the purpose of a yield statement?

Ans4

The yield statement is used in to define generator functions, which are functions that generate a sequence of values on-the-fly instead of creating them all at once like a list. When a generator function is called, it returns a generator object that can be iterated over using a for loop or the next() function.

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

Ans5

map() and list comprehensions are two ways to apply a function to a sequence of values.

map() is a built-in function that applies a given function to each element of a sequence and returns a new sequence with the results.

def square(x):

return x\*\*2

numbers = [1, 2, 3, 4, 5]

squares = map(square, numbers)

print(list(squares))

output

[1, 4, 9, 16, 25]

List comprehensions are a construct that allow you to create a new list by applying an expression to each element of an existing sequence.

numbers = [1, 2, 3, 4, 5]

even\_squares = map(lambda x: x\*\*2, filter(lambda x: x % 2 == 0, numbers))

print(list(even\_squares))

output

[4, 16]