* Getting Started with generators
* Pre-built generators
* Advantages of using generators
* Developing custom generators
* Exercise and Solution

1. What is the purpose of the yield statement in Python?

a) To return a value from a function

b) Returns a generator object and produce a series of values that can be iterated.

c) To pause the execution of a function and return a value to the caller

d) To throw an exception and stop the execution of a function

Answer: b

1. Which of the following is an advantage of using generators in Python?

a) They require less memory than lists

b) They can be used as a replacement for loops

c) They can be modified after creation

d) They can hold any type of data, including functions

Answer: a

1. What happens when you call a generator function in Python?

a) It returns a list of values

b) Returns a generator object.

c) It executes the function's code once and then stops

d) It raises a StopIteration exception

Answer: b

1. What is the difference between a generator expression and a list comprehension in Python?

a) A generator expression is evaluated lazily, while a list comprehension is evaluated eagerly

b) A generator expression creates a list, while a list comprehension creates a generator

c) A list comprehension is more memory-efficient than a generator expression.

d) A generator expression can be used as an iterator, while a list comprehension cannot

Answer: C

1. What is the "count" generator in the itertools module used for?

a) To generate an infinite sequence of numbers

b) To generate a sequence of numbers up to a certain limit

c) To generate a sequence of random numbers

d) To generate a sequence of Fibonacci numbers

Answer: a

1. What is the "chain" generator in the itertools module used for?

a) To combine multiple iterables into a single sequence

b) To generate a sequence of values based on a mathematical formula

c) To generate a sequence of values based on a conditional statement

d) To generate a sequence of values that are sorted in ascending order

Answer: a

1. How do generators save memory in Python?

a) By storing all their values in a list

b) By computing values on demand instead of storing them in memory

c) By using a smaller data type for their values

d) By using a compressed format to store their values

Answer: b

1. What is the main disadvantage of using generators in Python?

a) They can only be used with numerical data types

b) They can be slower than traditional loops for small datasets

c) They require more memory than lists

d) They are one-time iterables and cannot be reset to its initial state.

Answer: d

1. How do you define a coroutine in Python?

a) Using the "def" keyword and the "yield" statement

b) Using the "def" keyword and the "async" keyword

c) Using the "class" keyword and a "coroutine" method

d) Using the "yield from" statement

Answer: b

1. What is the purpose of the "send" method in a coroutine?

a) To send values back to the coroutine from the caller

b) To start or stop the coroutine

c) To pause the coroutine and return a value to the caller

d) To generate an exception and stop the coroutine

Answer: a