Module - 1

(1) How memory is managed in Python?

🡪 Python's garbage collector automatically manages memory by identifying and reclaiming unused objects.

Key aspects of Python's memory management:

(a) Automatic Memory Management:

🡪 Python uses automatic memory management, which means you don’t have to manually allocate and free memory. Instead, the python interpreter handles it for you.

(b) Reference Counting:

🡪 Python keeps track of the number of references to an object.

🡪 When the reference count becomes zero (no objects are pointing to it), the object is considered garbage and is reclaimed.

(c) Garbage Collection:

🡪 Python's garbage collector periodically scans memory for unused objects and reclaims them. It also detects cycles and reclaims objects involved in them.

(d) Memory Pools:

🡪 To improve efficiency, Python uses memory pools to allocate and deallocate small objects.

🡪 This reduces the overhead of individual memory allocation and deallocation operations.

(e) Object Allocation:

🡪 When an object is created, Python allocates memory from a suitable memory pool.

🡪 The reference count for the object is initialized to zero.

(f) Object Deallocation:

🡪 When an object's reference count reaches zero, it is reclaimed.

(2) What is the purpose continue statement in python?

🡪 The continue statement in Python is used within loops (like for or while) to skip the current iteration and move to the next one. When continue is encountered, the rest of the code inside the loop for that iteration is ignored, and the loop proceeds with the next iteration.

for number in range(5):

if number == 2:

continue # Skip the number 2

print(number)

Output:

0 1 3 4

(3) What are negative indexes and why are they used?

🡪 Negative indexes in Python are a powerful feature that allows us to access elements in a list from the end instead of the beginning.

Use: List, Tuple and string.