

1. Why might you choose a deque from the collections module to implement a queue instead of using a regular Python list?

A deque (double-ended queue) from the collections module is preferred for implementing a queue because it allows fast insertion and deletion from both ends. In a queue, we frequently remove elements from the front. In a Python list, removing from the front using `pop(0)` is slow because it shifts all elements ($O(n)$). But `deque.popleft()` removes elements from the front in constant time ($O(1)$), making it more efficient for queue operations.

2. Can you explain a real-world scenario where using a stack would be a more practical choice than a list for data storage and retrieval?

A real-world example of a stack is the **browser back button**. When a user opens multiple web pages, the last opened page should be the first one to return to when pressing “Back.” This follows the LIFO (Last In, First Out) rule, which is exactly how a stack works. Another example is **Undo/Redo operations** in text editors, where the most recent action is undone first.

3. What is the primary advantage of using sets in Python, and in what type of problem-solving scenarios are they most useful?

The primary advantage of a set in Python is that it stores **unique elements only** and provides **very fast membership checking** (like searching). Sets are most useful in scenarios such as removing duplicate values from data, finding common elements between two collections, and performing operations like union, intersection, and difference (e.g., finding common students in two classes).

4. When might you choose to use an array instead of a list for storing numerical data in Python? What benefits do arrays offer in this context?

An array is preferred over a list when storing a large amount of numerical data of the same type (such as integers or floats). Arrays consume less memory and can be faster for numeric processing because they store data in a more compact form. Arrays are commonly used in scientific computing, data analysis, and applications involving large numeric calculations. (NumPy arrays are widely used for this purpose.)

5. In Python, what’s the primary difference between dictionaries and lists, and how does this difference impact their use cases in programming?

The primary difference is that a list stores elements in an ordered sequence and is accessed using indexes (position-based), whereas a dictionary stores data in key-value pairs and is accessed using keys (name-based). Lists are useful when data is ordered and accessed by position, while dictionaries are best when data needs meaningful labels and fast lookup, such as storing student details like name, roll number, and marks.