Theoretical Questions (Data Types and Structures)

1) What are data structures, and why are they important?

Data structures are ways of organizing and storing data efficiently. They help with fast access, modification, and are essential for algorithms.

2) Difference between mutable and immutable data types

Mutable → changeable (list, dict, set). Immutable → unchangeable (int, str, tuple).

3) Differences between lists and tuples

List: mutable, slower, not hashable. Tuple: immutable, faster, hashable.

4) How dictionaries store data

They use a hash table internally with key-value pairs.

5) Why use a set instead of a list

Sets remove duplicates and allow fast membership checks.

6) What is a string and difference from list

String = immutable sequence of chars. List = mutable sequence of objects.

7) How tuples ensure data integrity

They are immutable, preventing accidental modification.

8) What is a hash table & relation to dict

Hash table stores keys/values via hash function. Python dicts use this.

9) Can lists contain different data types?

Yes, e.g. [1,'a',3.14].

10) Why strings are immutable

For efficiency, security, thread safety.

11) Advantages of dict over list

Dictionaries allow fast key lookups (O(1)).

12) Scenario for tuple use

Coordinates (x,y) or as dict keys.

13) How sets handle duplicates

They automatically remove duplicates.

14) `in` for list vs dict

List: checks values. Dict: checks keys.

15) Modify tuple elements?

No, tuples are immutable.

16) What is a nested dictionary

Dict inside a dict. Example: student with 'grades' dict.

17) Time complexity accessing dict

Average O(1), worst O(n).

18) When lists preferred over dicts

When order or duplicates matter.

19) Why dict unordered

Conceptually not indexed, retrieval is by key.

20) Difference list vs dict retrieval

List: by index. Dict: by key.