Python Theory Assignment

Q1. Understanding how to create and access elements in a list.

A list in Python is an ordered and mutable collection of elements. Lists are created using square brackets []. They can store multiple data types such as integers, floats, strings, and other objects. Elements are accessed using indexing which starts from 0. Lists are widely used for storing and managing collections of data.

Q2. Indexing in lists (positive and negative indexing).

Positive indexing starts from 0 and moves left to right, while negative indexing starts from -1 and moves right to left. Indexing allows direct access to list elements and helps in updating or retrieving values easily.

Q3. Slicing a list: accessing a range of elements.

Slicing is used to access a subset of list elements. Syntax is list[start:end]. The start index is inclusive and end index is exclusive. Slicing is useful for working with portions of lists.

Q4. Common list operations: concatenation, repetition, membership.

Concatenation joins lists using + operator. Repetition repeats list using * operator. Membership checks element existence using in keyword.

Q5. Understanding list methods append(), insert(), remove(), pop().

append() adds element at end, insert() adds at specific index, remove() deletes by value, pop() deletes by index.

Q6. Iterating over a list using loops.

Lists can be iterated using for loop to access each element one by one.

Q7. Sorting and reversing a list using sort(), sorted(), and reverse().

sort() sorts list permanently, sorted() returns new sorted list, reverse() reverses list order.

Q8. Basic list manipulations: addition, deletion, updating, slicing.

Lists support adding, deleting, updating, and slicing elements, making them dynamic.

Q9. Introduction to tuples and immutability.

A tuple is an ordered but immutable collection. Once created, elements cannot be changed.

Q10. Creating and accessing elements in a tuple.

Tuples are created using parentheses and accessed using indexing similar to lists.

Q11. Basic operations with tuples.

Tuples support concatenation, repetition, and membership operations.

Q12. Slicing a tuple to access ranges of elements.

Tuple slicing allows accessing a range of elements without modification.

Q13. Introduction to dictionaries: key-value pairs.

A dictionary stores data as key-value pairs. Keys are unique and used to access values.

Q14. Accessing, adding, updating, and deleting dictionary elements.

Dictionary elements can be accessed using keys and modified easily.

Q15. Dictionary methods like keys(), values(), and items().

These methods return keys, values, and key-value pairs respectively.

Q16. Iterating over a dictionary using loops.

Dictionaries can be iterated using for loops over keys, values, or items.

Q17. Merging two lists into a dictionary using loops or zip().

Two lists can be merged into a dictionary using loops or zip() function.

Q18. Counting occurrences of characters in a string using dictionaries.

Dictionaries are used to count frequency of characters efficiently.

Q19. Defining functions in Python.

Functions are reusable blocks of code defined using def keyword.

Q20. Different types of functions and lambda functions.

Functions can have parameters or return values. Lambda functions are anonymous functions.

Q21. Introduction to Python modules and standard modules.

Modules are files containing Python code. math and random are standard modules.

Q22. Creating custom modules.

Custom modules are user-defined Python files used for code reuse.