**Python Lists, Sets, Dictionaries, and Sequence Comparisons**

**Lists**

List Methods: Detailed explanation of various list methods, including append (), extend (), insert (), remove (), pop(), clear(), index(), count(), sort(), reverse(), and copy().

Using Lists as Stacks and Queues: Explains how list methods make lists usable as stacks (last-in, first-out) and demonstrates the usage of append () and pop () methods.

List Comprehensions: Shows how to use list comprehensions to create new lists based on expressions and various conditional clauses.

Nested List Comprehensions: Explores the utilization of list comprehensions within other list comprehensions, demonstrating 3x4 matrix transposition.

Daily Notes - The del Statement / Tuples and Sequences

List Operations: Describes the del statement for removing elements or entire variables from lists.

Sequence Data Types: Explains the common properties of lists and strings and introduces tuples as an immutable sequence data type.

Tuple Introduction and Properties: Highlights the immutability of tuples and their potential to hold mutable objects.

Differences Between Lists and Tuples: Clarifies the differences in mutability, element types, and access between lists and tuples.

Special Cases in Tuple Creation: Explains creating empty and single-item tuples with parentheses.

Tuple Packing and Unpacking: Demonstrates tuple packing and unpacking for assignments and sequence data manipulation.

**Sets**

Sets: Introduces sets as an unordered collection of unique elements, showcasing the usage of set creation, operations, and set comprehensions.

Dictionaries: Describes dictionaries as key-value pairs, dict initialization, operations like storing, extracting, and deleting key-value pairs, and various dictionary construction methods.

Daily Notes - Looping Techniques

Looping through Dictionaries: Demonstrates how the items () method works for dictionary key-value retrieval.

Looping through Sequences: Introduces functions like enumerate (), zip (), and demonstrates their usage in looping through sequences.

Looping in Reverse and Sorted Order: Explains how to iterate over sequences in reverse and sorted orders.

Best Practices for Modifying Lists: Emphasizes the potential issues and recommends creating new lists instead of modifying existing lists while iterating.

**More on Conditions**

Boolean Expressions in Python: Explores the flexibility of using various operators in while and if statements.

Comparison Operators: Discusses a wide range of comparison operators like in, not in, is, is not, and their different functionalities.

Chained Comparisons and Boolean Operators: Covers chaining comparisons and using Boolean operators like and, or, and not.

Short-Circuit Evaluation and Assignment: Details the behaviour of short-circuit operators and the assignment of comparison results to variables.

Python's Assignment and Comparison: Highlights the prevention of errors by not allowing assignments within expressions.

**Comparing Sequences and Other Types**

Lexicographical Ordering: Explains how lexicographical ordering compares sequences, recursively for similar sequence types.

Examples of Comparisons: Provides examples of comparing tuples, lists, strings, etc., based on lexicographical order.

Handling Different Object Types in Comparisons: Demonstrates the behaviour of comparing objects of different types and how the interpreter raises exceptions in cases where no logical comparison can be made.