Lab # 7

Chapter 10 • introduces Python lists as flexible and powerful data structures that can store multiple items, such as numbers, strings, or even other lists. Lists are created using square brackets "[]", and elements can be accessed using index values starting at zero. Common operations include checking length with "len()", determining membership with "in" keyword, and accessing slices of the list. Lists are mutable, meaning their content can be changed after creation, unlike strings or tuples. The chapter also explains how lists are referenced in memory, so assigning one list another variable doesn't create a copy — it points to the same data. To avoid unintended changes, cloning with slicing ([:]) is recommended. The book also explores list methods like ".append(), .insert(), and .remove()," as well as how to use "for loops" to iterate through list items. A key pattern discussed is the accumulator pattern, where a variable is updated while looping to compute totals, find maximums, or build new lists. Functions can receive and return lists, and Pythons list comprehensions offer a concise way to create lists using simple syntax. Toward the end, the chapter covers nested lists, the "list()" type conversion, and introduces tuples — immutable list-like structures often used to return multiple values from functions. Overall, the chapter provided a comprehensive foundation for understanding how to manipulate and use lists effectively in Python.