ASSIGNMENT - 13

Q1. Can you create a programme or function that employs both positive and negative indexing? Is there any repercussion if you do so?

Ans: def index\_example(lst):

# Positive indexing

print("First element:", lst[0])

# Negative indexing

print("Last element:", lst[-1])

Employing both positive and negative indexing in a program won't lead to any issues as long as the indexes are within the range of the list. Negative indexes count backward from the end (-1 being the last element), while positive indexes start from 0 at the beginning.

Q2. What is the most effective way of starting with 1,000 elements in a Python list? Assume that all elements should be set to the same value.

Ans: The most effective way to start with 1,000 elements in a Python list, all set to the same value, is by using list comprehension:

# Creating a list with 1,000 elements, all set to the same value (e.g., 0)

my\_list = [0] \* 1000

Q3. How do you slice a list to get any other part while missing the rest? (For example, suppose you want to make a new list with the elements first, third, fifth, seventh, and so on.)

Ans: To create a new list with alternate elements (e.g., first, third, fifth, and so on), you can use slice notation with a step size:

original\_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

new\_list = original\_list[::2] # Extracting elements with a step of 2

print(new\_list) # Output: [1, 3, 5, 7, 9]

Q4. Explain the distinctions between indexing and slicing.

Ans: Distinctions between Indexing and Slicing:

* Indexing retrieves a single element at a specific position in the list.
* Slicing retrieves a portion (sublist) of the list by specifying a range of indices (start:end:step).

Q5. What happens if one of the slicing expression’s indexes is out of range?

Ans: If one of the slicing expressions' indexes is out of range, Python will handle it gracefully. It won't raise an error but will return the slice that is within the valid range.

Q6. If you pass a list to a function, and if you want the function to be able to change the values of the list—so that the list is different after the function returns—what action should you avoid?

Ans: To enable a function to change the values of a list passed to it, avoid reassigning the entire list to a new object within the function. Instead, modify the list in place using its methods or by directly accessing its elements.

Q7. What is the concept of an unbalanced matrix?

Ans: An unbalanced matrix is a matrix that doesn't have an equal number of rows and columns. In such matrices, the number of elements in each row may differ, causing an imbalance in the structure.

Q8. Why is it necessary to use either list comprehension or a loop to create arbitrarily large matrices?

Ans: List comprehension or loops are necessary to create arbitrarily large matrices because they allow dynamic generation of elements. Directly initializing a large matrix without comprehension or loops would require manual typing or predefined patterns, which becomes impractical as the size of the matrix increases. Using loops or comprehensions enables the creation of large matrices efficiently and dynamically.