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

Ans1- Yes, you can use both positive and negative indexing in a Python program or function. Positive indexing accesses elements from the beginning, while negative indexing accesses elements from the end.

There's no issue with using both, but you should be aware that using negative indexing may lead to unexpected behavior if you don't handle it correctly, such as accessing elements beyond the list's length.

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.

Ans2- The most effective way to create a list with 1,000 elements, all set to the same value, is to use a list comprehension or the \* operator.

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.)

Ans3- To create a new list with alternate elements from an existing list, you can use slicing with a step value of 2.

List1 = [1, 2, 3, 4, 5, 6, 7, 8, 9]

List2 = list1[::2]

Q4. Explain the distinctions between indexing and slicing.

Ans4- Indexing accesses a single element at a specific position in a list or sequence.

Slicing extracts a portion of a list or sequence, creating a new list with elements from a specified range.

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

Ans5- If one of the slicing expression's indexes is out of range, Python doesn't raise an error. Instead, it silently truncates or extends the slice to fit within the valid index 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?

Ans6- To avoid a function mutating a list passed as an argument, you should avoid modifying the list directly within the function. Instead, you can create a copy of the list inside the function and make changes to the copy.

Q7. What is the concept of an unbalanced matrix?

Ans7- An unbalanced matrix is a matrix where the number of rows and the number of columns are not equal. It means the matrix is not square, and it may have different dimensions for rows and columns.

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

Ans8- List comprehension or loops are necessary to create arbitrarily large matrices in Python because they allow you to efficiently generate and populate the matrix elements based on a pattern or formula.