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

Def indexing(start\_ind,end\_ind,string\_text):

If start\_ind >0 & end\_ind>0:

Return string\_text[start\_ind:end\_ind]

Else:

Return string\_text[-start\_ind:-end\_ind]

Yes there are the repercussion of this by postivie and negative indexing will give the total different substring based on the index which we are providing.

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.

[i\*0 for i in range(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.)

ll= list1[::2]

By adding the step size as given above we can achieve the same.

Q4. Explain the distinctions between indexing and slicing.

Indexing means accessing the single elements from the string / list by giving the index.

With slicing we can access the range of the index like by mentioning the starting index and the end index.

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

If the upper index is out of range then we will still get the complete output.

In case if both the lower and upper indexes are out of range then it will give an output as empty string / list.

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?

Instead of printing a list we need to return the updates list from the function.

We need to assign the operation whatever we are performing to the list.

Q7. What is the concept of an unbalanced matrix?

A matrix is balanced if all cells in the matrix are balanced and a cell of the matrix is balanced if the number of cells in that matrix that are adjacent to that cell is strictly greater than the value written in this cell.

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

These are the compacts techniques through which we can generate the matrices with minimum number lines of codes in it.

Also the code which is written with loop and list comprehension are easy readable.