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

Ans # positive indexing

def positive\_indexing(l):

pos=l[2]

return pos

positive\_indexing([11,22,33,44,55,66,77])

# negative indexing

def negative\_indexing(l):

neg=l[-2]

return neg

negative\_indexing([11,22,33,44,55,66,77])

The penalty for doing positive and negative is, positive will count the index position from left side and starts with 0 position.

While in negative indexing, counting is from the right and start with -1

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 By using list comprehension operation we can acheive this

l=[i for i in range(1,1001)]

print(l)

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 l= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

l[::2]

Q4. Explain the distinctions between indexing and slicing.

Ans Indexing refers to an element of an iterable by its position within the iterable.

#Slicing means getting a subset of elements from an iterable based on their indexes.

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

Ans l= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

If start index is out of range then it will return empty entity. If end index is out of range then it will return the items upto last index position form the start position mention

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 def func(l):

l[1]=100

return l

func([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20])

we can easily change the value of the item inside the list by just definig some function. So instead of writing another different list from the scratch and then pass it to the function can be avoid

Q7. What is the concept of an unbalanced matrix?

Ans Unbalanced Matrix is a matrix in which number of rows are not equal to number of columns. Like 8 x 5 is a unbalance matrix but 8 x 8 is a balanced matrix

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

Ans Because if we do manually by writing every item then firstly this will take a huge time, secondly memory will occupy more, thirdly it is not that much readable to the user and not better understanding. So thats why we use either looping or list comprehension so that in a very short way way we can create a large matrix