1. What exactly is []?

A: list

2. In a list of values stored in a variable called spam, how would you assign the value 'hello' as the third value? (Assume [2, 4, 6, 8, 10] are in spam.)

A: spam = [2,4,6,8,10]

print('Before insert', spam)

spam.insert(2,'Hello')

print('after insert', spam)

Let's pretend the spam includes the list ['a', 'b', 'c', 'd'] for the next three queries.

3. What is the value of spam[int(int('3' \* 2) / 11)]?

A: 2

4. What is the value of spam[-1]?

A: ’d’

5. What is the value of spam[:2]?

A: ‘c’

Let's pretend bacon has the list [3.14, 'cat,' 11, 'cat,' True] for the next three questions.

6. What is the value of bacon.index('cat')?

A: 1

7. How does bacon.append(99) change the look of the list value in bacon?

A: [3.14, 'cat', 11, 'cat', True, 99].

Since append function always adds the value in the end, hence 99 was placed in the end.

8. How does bacon.remove('cat') change the look of the list in bacon?

A: it will remove ‘cat’ of index[1] therefore, the output will be :

[3.14, 11, 'cat', True, 99]

9. What are the list concatenation and list replication operators?

A: In concatenation, two different /same string of different variables joins whereas, in replication operators, same string concatenate the number of times mentioned.

Concatenate operator: +

Replication operator: \*

Example:

a = ‘Navtej’

b = ‘Nimmi’

print(a+b) #concatenation #output: NavtejNimmi

a = ‘Navtej’\*2

print(a) #replication #output: NavtejNavtej

10. What is difference between the list methods append() and insert()?

A: append(), adds the value at the end whereas, insert() we can place the value to desired index value.

11. What are the two methods for removing items from a list?

A: remove()

pop()

12. Describe how list values and string values are identical.

A: We can access each element of string or list like string[0]/list[1]

13. What's the difference between tuples and lists?

A: lists are mutable but tuples are not

14. How do you type a tuple value that only contains the integer 42?

A: tup = (42,)

15. How do you get a list value's tuple form? How do you get a tuple value's list form?

A: ref code: lis = [1,2,3,4]

tup = tuple(lis)

or

tup = (1,2,3,4)

lis = list(tup)

16. Variables that "contain" list values are not necessarily lists themselves. Instead, what do they contain?

A: reference to the list object in the memory

17. How do you distinguish between copy.copy() and copy.deepcopy()?

A: both the fucntions performs copy however, copy.copy() allows shallow copy meaning, if we modify original list then copied list will be automatically modified. In case of deep.copy(), it creates an independent element meaning, if we modify original list then copied list will not be updated.