1. What exactly is []?

[] are used to decare a list in python

[] is also used in list comprehension to perform operations on items in list

1. 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.)

Spam[2] = ‘hello’

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

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

‘d’

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

‘d’

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

[‘a’, ‘b’]

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

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

1

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

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

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

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

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

L1.extend(l2)

L1 \* 2

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

Append() adds element at the end of the list

Insert(index, value) inserts at a given index if index not given throws error

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

List.pop() removes last element

List.remove(element) removes the first occurrence of element in list

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

String is nothing but list of characters

Both list and string stores indexed elements

One is mutable another is iimutable

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

Tuple List

Immutable mutable

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

Tup = (42,)

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

list(tup)

tuple(bacon)

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

Reference to the value in list

>>> bacon

['b', 'c', 'a']

>>> LETTER = bacon[0]

>>> LETTER

'b'

>>> id(bacon[0])

1517249352560

>>> id(LETTER)

1517249352560

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

L1 = [list, int, string ]

L2 = copy.copy(l1) creates l2 with [list, int, string ]

In case of int and string it does not cause any problem as it copies the value of the elements but for l1[0] copy.copy() copies the address of list inside the list hence both l1[0] and l2[0] refer to the same list

Hence any alteration in that list will affect both l1 and l2

Deep.copy() helps avoid this as it creates a new list and assigns its referece in l2[0]

>>> import copy

>>> l1 = [[1,2,3],4,5,6,7,8]

>>> l\_s = copy.copy(l1)

>>> l\_d = copy.deepcopy(l1)

>>>

>>> l1[0][0] = 'a'

>>> l\_s

[['a', 2, 3], 4, 5, 6, 7, 8]

>>> l\_d

[[1, 2, 3], 4, 5, 6, 7, 8]

>>> l1

[['a', 2, 3], 4, 5, 6, 7, 8]