1. What exactly is []?

A. An empty 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] = 'hello'

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. Answer is 'd'. As the '3' \* 2 = 33 then 33/11 = 3 and spam[3] is d

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

A. 'd'

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

A. ['a', 'b']

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]

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

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

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

A. + is used for list concatenation and \* is used as list replication operator.

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

A. append() adds an element to the end of the list. insert() inserts an element at a specified position in the list, shifting the existing to right.

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

A. pop, remove

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

A. Both are sequences of elements. Both support indexing, slicing, iteration, and membership testing.

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

A. lists are mutable and use square brackets, while tuples are immutable and use parentheses.

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

A. my\_tuple = (42,)

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

A. I will convert list to tuple for getting tuple form and convert tuple to list for getting list form.

eg:

t = tuple([1,2,3])

l = list((1,2,3))

print(t,l)

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

contain?

A. They contain references or pointers to the list objects.

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

A. The copy.copy() and copy.deepcopy() are the functions of copy module and are used to create copies of objects, but they are different in handling nested or referenced objects.

eg:

import copy

original\_list = [1, [2, 3], 4]

# Shallow copy

shallow\_copy = copy.copy(original\_list)

original\_list[1].append(5)

# Deep copy

deep\_copy = copy.deepcopy(original\_list)

print("Original List:", original\_list) # Output: [1, [2, 3, 5], 4]

print("Shallow Copy:", shallow\_copy) # Output: [1, [2, 3, 5], 4]

print("Deep Copy:", deep\_copy) # Output: [1, [2, 3], 4]