**1. What exactly is []?**

[] represents an empty list which does not have any elements in it.

**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.)**

**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)]?**

‘d’

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

‘d’

**5. 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.**

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

1

**7. 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?**

List Concatenation:

* ‘+’ operator

List Replication:

* ‘\*’ operator

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

append() is used to add new elements in the list.

insert() is used to modify already existing elements in a list.

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

remove() and pop()

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

The similarity between Lists and Strings values in Python is that both are sequences.

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

Tuples are immutable whereas lists are mutable in nature.

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

tuple = (42)

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

l = [4,5,6]

tuple(l)

l = (1,2,3)

list(l)

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

Those variables contain the references to the list values.

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

If we make changes to a list created using copy.copy(), it will make changes in the original list also as the elements are passed by references.

If we make changes to a list created using copy.deepcopy(), it will not make any changes in the original list also as the elements are passed by value.