1. What exactly is []?

**Answer:** 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.)

**Answer:** spam[3]=”hello”

[2, 4, 6, 'hello', 10]

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)]?

**Answer:** d

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

**Answer:** d

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

**Answer:** ['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')?

**Answer:** 1

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

**Answer:** [3.14, 'cat', 11, 'cat', True, 99]

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

**Answer:** [3.14, 11, 'cat', True, 99]

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

**Answer:** + forConcatenation

\* for replication

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

**Answer:** append() always adds the element at the end of the list

Insert() adds element in between the already existing elements in the list at a specified index

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

**Answer:** list.remove(), list.pop()

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

**Answer:** Both can be accessed with their index values

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

**Answer:** The tuples are immutable objects and the lists are mutable

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

**Answer:** tuple=(42)

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

**Answer:** tuple(list[]), list[tuple()]

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

**Answer:** Variables will contain references to list values rather than list values themselves. But for strings and integer values, variables simply contain the string or integer value.

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

**Answer: 1.** In case of deep copy, a copy of object is copied in other object. It means that any changes made to a copy of object do not reflect in the original object. In python, this is implemented using “deepcopy()” function.

**2.** In case of shallow copy, a reference of object is copied in other object. It means that any changes made to a copy of object do reflect in the original object. In python, this is implemented using “copy()” function.