1. What is []?

**Answer:** The empty list value, which is a list value that contains no items.

2. How would you assign the value 'hello' as the third value in a list stored in a variable named spam? (Assume spam contains [2, 4, 6, 8, 10].)

**Answer:** spam=[2,4,6,8,10]

Spam[2]=”hello”

Spam

**Output:** [2,4,”hello”,8,10]

For the following three questions, let’s say spam contains the list ['a', 'b', 'c', 'd'].

3. What does spam[int(int('3' \* 2) // 11)] evaluate to?

**Answer:** ‘d’

4. What does spam[-1] evaluate to?

**Answer:** ‘d’

5. What does spam[:2] evaluate to?

**Answer:** [‘a’,’b’]

For the following three questions, let’s say bacon contains the list [3.14, 'cat', 11, 'cat', True].

6. What does bacon.index('cat') evaluate to?

**Answer:** bacon.index(‘cat’)

**Output: 1**

7. What does bacon.append(99) make the list value in bacon look like?

**Answer:** bacon.append(99)

Bacon

**Output:** [3.14, ‘cat’, 11, ‘cat’, True, 99]

8. What does bacon.remove('cat') make the list value in bacon look like?

**Answer:** bacon.remove(‘cat’)

Bacon

**Output:** [3.14, 11, ‘cat’, True, 99]

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

**Answer:** The operator for list concatenation is **+,** while the operator for replication is **\*.**

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

**Answer:** The only difference between **append()** and **insert()**  is the **insert()** function allows us to add a specific element at a specified index of the list unlike **append()** where we can add the element only at end of the list

11. What are two ways to remove values from a list?

**Answer:** 1. Using the **remove()** method.

2. Using the list object’s **pop()** method.

3. Using the **del** operator.

12. Name a few ways that list values are similar to string values.

**Answer:** Both **lists** and **strings** can be passed to len(), have indexes and slices, be used in for loops, be concatenated or replicated, and be used with the in and not in operators.

13. What is the difference between lists and tuples?

**Answer:** The most important **differences between list and tuple** is that **list** is mutable, whereas a **tuple** is immutable. This means that lists can be changed, and **tuples** cannot be changed. But **tuples** are immutable, they cannot be copied.

14. How do you type the tuple value that has just the integer value 42 in it?

**Answer:** (42,) the trailing comma is mandatory.

15. How can you get the tuple form of a list value? How can you get the list form of a tuple value?

**Answer:** Python **list** of **tuples** using **list** comprehension and **tuple()** method. Python **tuple()** method along with **List** comprehension can be used to form a list of tuples. The tuple() function helps to create tuples from the set of elements passed to it.

16. Variables that “contain” list values don’t actually contain lists directly. What do they contain instead?

**Answer:** They contain **references** to list values.

17. What is the difference between copy.copy() and copy.deepcopy()?

**Answer: deepcopy()** creates new object and does real **copying** of original object to new one. **Deepcopy() copies** original object recursively, while . **copy()** create a reference object to first-level data of original object. So the **copying/referencing** difference between.