1. What exactly is []?

**Ans**.

* It is an empty list container.
* It does not contain any element in it.
* It is used to initialise an empty list which can be updated later.

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.)

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

**Ans.** spam[2] = ‘hello’

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

**Ans.**

As, spam = ['a', 'b', 'c', 'd']

spam[int(int('3' \* 2) / 11)] 🡪 spam[33/11]=spam[3]

It will return value **‘d’**.

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

**Ans.**

It will return **last element** which is **‘d’**

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

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

**Ans.**

Spam[:2] = spam[0:2] where 0th index is inclusive and 2nd index is exclusive.

Its value = [‘a’ , ‘b’]

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

**Ans.**

As, bacon = [3.14, 'cat,' 11, 'cat,' True]

.index(element) 🡪 It will return the index of an element in list.

It will return **‘1’** … (index of first occurrence of ‘cat’)

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

**Ans.**

.append(data) will append the data at the end of list.

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

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

**Ans.**

bacon = [3.14, ' 11, 'cat,' True, 99] … (removed first occurrence of ‘cat’)

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

**Ans.**

**List concatenation operator: +**

**e.g.** [1,2] + [3,4] = [1,2,3,4]

**List replication operator: \***

**e.g.** 4\*[1] = [1,1,1,1]

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

**Ans.**

**.append(element) 🡪**

* It appends the data at the end of list.
* It takes one parameter as element to add it at last.

**.insert(index , element)** 🡪

* It inserts an element at specified value ‘index’.
* It takes two parameters as index and element.

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

**Ans.**

1. .remove(element)
2. .pop(index) … default last

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

**Ans.**

* Concatenation/replication of both lists and strings is possible.
* Both can be used with the ‘in’ and ‘not in’ operators.
* Both have indexes and slices.
* len() functions can be used with both.

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

**Ans.**

**Tuples:** They are immutable. Tuples can be represented in round bracket ( ).

**Lists:** They are mutable. Lists can be represented in square bracket [ ].

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

**Ans.**

T = (42,)

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

**Ans.**

**List value’s tuple form:**

List = [1,2,3,4]

tuple\_form = tuple(list)

tuple\_form = (1,2,3,4)

**tuple value's list form:**

tuple = (1,2,3,4)

list\_form = list(tuple)

list\_form = [1,2,3,4]

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

**Ans.** They contain **references** to list values.

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

**Ans.**

**copy.copy():** returns a shallow copy of a list.

**copy.deepcopy():** returns a deep copy of a list