1. What exactly is []?

**Answer:** In order to initialize a list we use [], these [] braces indicates an empty list.

A=[] ; indicates empty list

A=[1,2]; indicates a list of two parameters.

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[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)]?

**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:**  The list [3.14, 'cat,' 11, 'cat,' True] don’t have quotes in proper places if not corrected it will throw exception.

Post correcting the list to list [3.14, 'cat', 11, 'cat', True] the value of bacon.index('cat') will be 1.

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

**Answer:**  The list [3.14, 'cat,' 11, 'cat,' True] don’t have quotes in proper places if not corrected it will throw exception.

Post correcting the list to list [3.14, 'cat', 11, 'cat', True] the value of bacon.append(99) will be[ 3.14, 'cat', 11, 'cat', True,99]

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

**Answer:**  The list [3.14, 'cat,' 11, 'cat,' True] don’t have quotes in proper places if not corrected it will throw exception.

Post correcting the list to list [3.14, 'cat', 11, 'cat', True] the value of bacon.remove('cat') will be[ 3.14, 11, 'cat', True,99]

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

**Answer:** List Concatenation operator is + whereas List replication operator is \*.

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

**Answer: Append** operator will add the value at the end of the list whereas the **Insert** operator will add the values at the specified index of a list.

Suppose a = [1,2,3]

a.append(4) 🡪 [1, 2, 3, 4]

a.insert(0,5) 🡪 [5, 1, 2, 3, 4] 🡪 5 is inserted at the 0th Index of the list.

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

**Answer:** Two methods for removing items from a list: Remove() and Pop().

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

**Answer:** Both list and string values have sequences, both are iterable in both we can do indexing and slicing.

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

**Answer:** The major difference between tuples and lists is tuples are immutable whereas the lists are mutable.

Tuple starts and ends with paranthesis () whereas lists starts and ends with []

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:**

List value's tuple form: tuple(list\_a)

Tuple value's list form: list(tuple\_a)

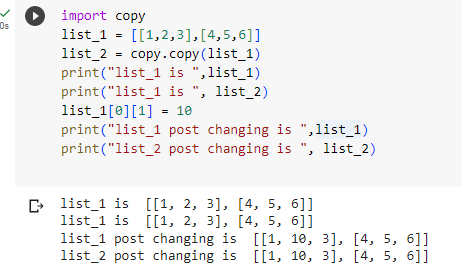
16. Variables that "contain" list values are not necessarily lists themselves. Instead, what do they contain?

**Answer:** Contains reference values of the other list.

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

**Answer:** copy.copy() and copy.deepcopy() both work same for non-nested lists, it makes a difference when we have a nested list.

**copy.copy()**copies all the items from list\_1 to list\_2 and if we change any item in list\_1 the changes will be reflected in list\_2 too, which implies copy.copy() makes a shallow copy.



Whereas **copy.deepcopy()**copies all the items from list\_1 to list\_2 and if we change any item in list\_1 the changes will not be reflected in list\_2, which implies copy. deepcopy () makes a deep copy basically whatever we do in both the list it gets stored in different memory locations thereby not disturbing each other.

