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

[] refers to List in python. It is one of the data structures which can hold different data types 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 = [2,4,6,8,10]

spam[2] = 'hello'

print(spam)

Out : [2, 4, 'hello', 8, 10]

*if we want to insert an element then*  
spam = [2,4,6,8,10]

spam.insert(2,'hello')

print(spam)

Out : [2, 4, 'hello', 6, 8, 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)]?**

‘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, 99]

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

List concatenation operator : +

Eg :

a = [1, 2, 3]

b = [4, 5, 6]

concatenated\_list = a + b

print(concatenated\_list)

Out: [1, 2, 3, 4, 5, 6]

-------------------------------------

List replication operator : \*

Eg :

main\_list = [1, 2, 3]

replicated\_list = main\_list \* 3

print(replicated\_list)

Out: [1, 2, 3, 1, 2, 3, 1, 2, 3]

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

append() method is used to add the element at the end of the list. It takes only one argument, i.e. the element that needs to be appended.

Eg :

list1 = [1, 2, 3]

list1.append(4)

print(list1)

Out : [1, 2, 3, 4]

insert() method is used to insert the element at the specified location in the list. It takes two arguments i.e. the index at which the element needs to be inserted and the element itself.

Eg :

List2 = [1, 2, 3]

List2.insert(1, 4)

print(List2)

Out: [1, 4, 2, 3]

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

pop() : it removes last item from the list of items, if we specify the index in the (1), it will remove that particular indexed element from the list.

Eg :

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

bacon.pop(1)

out : ‘cat’

remove() : it takes the element as argument to remove that specific element from the given list.

Eg :

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

bacon.remove(‘cat’)

print(bacon)

out : [3.14, 11, 'cat', True]

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

Both lists and strings are ordered collections of elements.

Both lists and strings support indexing and slicing.

We can iterate over both lists and strings using loops.

Lists are mutable but strings are immutable.

List can hold any kind of data types but strings can not.

Lists and strings have different methods and operations associated with them.

Eg: Lists have methods like append(), insert(), and remove().

Strings have methods for string manipulation,.

Eg: split(), join(), and replace(), which are specific to string operations.

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

Tuples are immutable, while lists are mutable.

Tuples are defined using () where lists are defined using []

Tuples generally have slightly better performance than lists because of their immutability.

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

Initiate a tuple with comma added.

Eg : tup1 = (42,)

Type(tup1)

Out : tuple

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

**To convert List to Tuple we use tuple() and pass list in to it.**

**Eg :**

list1 = [1,2,3,4,5]

tuple1 = tuple(list1)

out : (1,2,3,4,5)

**To convert Tuple to List we use list() and pass tuple in to it.**

**Eg :**

tup1 = (1,2,3,4,5)

list1 = list(tup1)

out : [1,2,3,4,5]

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

Python uses a concept called **reference semantics,** it means that a list containing elements is assigned to a variable. This variable will not hold the elements in the list instead it holds the reference to the memory location where the list is stored.

Variables will contain the reference to the memory location where the list is stored.

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

In copy module we have two copy options, copy.copy() and copy.deepcopy()

copy.copy() will creates a shallow copy. It creates a new object, but the elements are referred to the original object indices. Changing one element in original object will reflect in the copied object as well.

copy.deepcopy() will creates a new object and copies all the elements from the original object by iterating through it with new indices. So, changing the elements in any of the objects will not reflect on the other. Every object is individual to one another.