1. What exactly is []?

It is an empty list, used to initialize a 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.)

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)]?
Ans = d

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

5. What is the value of spam[:2]?
Ans = [‘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')?
Ans = 1

7. How does bacon.append(99) change the look of the list value in bacon?
Ans = [3.14, 'cat', 11, 'cat', True, 99]

8. How does bacon.remove('cat') change the look of the list in bacon?
Ans = [3.14, 'cat', 11, True, 99]

9. What are the list concatenation and list replication operators?
Ans = Concatenation adds the elements of two lists (example = A + B) while Replication replicates the elements of a list (example = A * 3)

10. What is difference between the list methods append() and insert()?
Ans = Appends add the element to the end of list while insert adds the element to a specified index of the list.

11. What are the two methods for removing items from a list?
Ans = pop() and remove()

12. Describe how list values and string values are identical.
Ans = Both are a list of iterable elements

13. What's the difference between tuples and lists?
Ans = Tuples are immutable (they cannot be changed), while lists can be changed.

14. How do you type a tuple value that only contains the integer 42?
Ans = (42,)

15. How do you get a list value's tuple form? How do you get a tuple value's list form?
a = [(1,2), (3,4)] (For tuple in a list)
b = ([2,4], [5,6]) (For list in a tuple)

16. Variables that "contain" list values are not necessarily lists themselves. Instead, what do they contain?
Ans = They will contain values of lists and the variable itself can be a list or tuple.

17. How do you distinguish between copy.copy() and copy.deepcopy()?
In case of copy.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 case of copy.deepcopy(), 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