1. What exactly is []?

**That is known as list which is mutable, usually of the same type**

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’, because first value starts with zero index. So, index 2 indicates third place in the list**

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)]?

**The value of spam[int(int(‘3’\*2)/11)] is ‘d’**

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

**The value of spam[-1] is ‘d’**

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

**The value of spam[:2] is [‘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')?

**The value of bacon.index(‘cat’) is 1**

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

**Bacon will be [3.14, ‘cat’, 11, ‘cat’, True,99]**

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

**Bacon will be [3.14,11,’cat’,True,99]**

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

**In insert we can add element at a specific index but in append we can add only at the end of list**

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

**First one is remove and another one is pop**

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

**Both are sequences**

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

**Lists are mutable while tuples are immutable objects**

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

**(42)**

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

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

**Variables will contain references to list values rather than list values themselves**

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

**copy() create reference to original object. If you change copied object - you change the original object. . deepcopy() creates new object and does real copying of original object to new one. Changing new deepcopied object doesn't affect original object.**