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

Empty list

1. **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.append('hello)

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

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

‘d’

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

‘d’

1. **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.

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

1

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

[3.14,’cat’,11,’True’,99]

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

bacon.pop(‘cat’)

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

The operator for list concatenation is +, while the operator for replication is \*

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

append() will add values only to the end of a list, insert() can add them anywhere in the list.

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

Pop(), remove()

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

Both have sequential, length, position

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

* List is mutable tuple is immutable
* List take much memory tuple take less memory
* Built in method are available in list but tuple do not have built in memory

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

Tup=(42)

1. **How do you get a list value's tuple form? How do you get a tuple value's list form?**
2. **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. But for strings and integer values, variables simply contain the string or integer value.

1. **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.