1. What exactly is []?

Ans: it is a 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.)

Ans: spam[2] = “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)]?

Ans: ‘d’

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

Ans: ‘d’

1. 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.

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

Ans: 1

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

Ans: [3.14, 'cat,' 11, 'cat', True, 99]

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

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

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

Ans: Concatenation is + while replication is \*.

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

Ans: append() add the element to the end of the list while insert() inserts the element to the specific index.

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

Ans: remove() and pop()

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

Ans: For both values we can iterate through it.

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

Ans: To define a tuple we use () while to define a list we use []. A tuple is immutable while lists are mutable objects.

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

Ans: t = (42,)

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

Ans: List value from tuple -> list()

Tuple value from list -> tuple()

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

Ans: Variables contains reference to list values.

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

Ans: copy.copy() creates a reference to the original object. Making changes to copied object will change the original also.

copy.deepcopy() creates new object and copy of the original object to new object. Making changes in new will not change the original object.