1. What exactly is []?

[] is used for List

L=[] will create an empty 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)]?

33/11=3

Spam[3]=d

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

Spam[-1]=d

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

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?

It adds 99 at the end of the list.

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

It removes the first occurrence of ‘cat’ and now the elements remain are [3.14, 11, 'cat', True, 99]

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

List concatenation: + is used to concatenate lists

e.g niti=[1,2,3]

sriv=[‘a’,’b’,c’]

niti+sriv:

[1,2,3,’a’,’b’,’c’]

List replication operator: \* it will repeat the list that number of times

Niti=[1,2,3]

Niti\*=2

Niti:

[1,2,3,1,2,3]

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

Append() will add the element to the end of the list

Insert() will add an element at the given particular index

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

Pop() and remove()

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

List and String both are identical.

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

List is mutable.

Tuple is immutable.

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

T=(42,)

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

List of tuples: [(tuple1), (tuple2), (tuple3), (tuple4), (tuple5) ]

Tuples of list: tuple( [list1], [list2] )

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

Those variables basically contain reference to the list

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

For both of these data should be some form of collection.

1. Copy.copy()- shallow copy

Eg. list1=[1,2,3,4]

list2=list1.copy()

list2[1]=1000

list2: [1, 1000, 3, 4]

list1: [1, 2, 3, 4]

in case of nested lists:

list1=[[1,2,3,4],[5,6,7,8]]

list2=list1.copy()

list1[1][0]=100

list1,list2:

([[1, 2, 3, 4], [100, 6, 7, 8]], [[1, 2, 3, 4], [100, 6, 7, 8]])

1. Deep copy

list1=[1,2,3,4]

list2=copy.deepcopy(list1)

list2[1]=1000

list1,list2:

([1, 2, 3, 4], [1, 1000, 3, 4])

In case of multi dimension list:

list1=[[1,2,3],[4,5,6]]

list2=copy.deepcopy(list1)

list2[0][1]=1000

list1:

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

list2:

[[1, 1000, 3], [4, 5, 6]]