1. What exactly is []?

[] is an 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.)

a=[2, 4, 6, 8, 10]

a.append('Hello')

print(a)

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)]?

spam=['a', 'b', 'c', 'd']

spam=spam[int(int('3' \* 2) / 11)]

print(spam)

o/p: 3

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

spam=['a', 'b', 'c', 'd']

print(spam[-1])

o/p: d

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

spam=['a', 'b', 'c', 'd']

print(spam[-2])

o/p: c

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')?

bacon = [3.14, 'cat', 11, 'cat',True]

bacon.index('cat')

o/p 1

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

bacon = [3.14, 'cat', 11, 'cat',True]

bacon.append(99)

print(bacon)

o /p [3.14, 'cat', 11, 'cat', True, 99]

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

bacon = [3.14, 'cat', 11, 'cat',True]

bacon.remove('cat')

print(bacon)

o/p [3.14, 11, 'cat', True]

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

Concatenation operator is used to concatenate two lists

bacon = [3.14, 'cat', 11, 'cat',True]

bacon+['aa']

o/p [3.14, 'cat', 11, 'cat', True, 'aa']

Replication operator is used to multiply the list by number of times

bacon = [3.14, 'cat', 11, 'cat',True]

bacon\*2

o/p: [3.14, 'cat', 11, 'cat', True, 3.14, 'cat', 11, 'cat', True]

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

The difference is that with append, you just add a new entry at the end of the list. With insert (position, new\_entry) you can create a new entry exactly in the position you want

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

Remove and pop are two methods to remove items from the list

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

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

|  |  |
| --- | --- |
| Lists are mutable | Tuples are immutable |
| Implication of iterations is Time-consuming | The implication of iterations is comparatively Faster |
| The list is better for performing operations, such as insertion and deletion. | Tuple data type is appropriate for accessing the elements |
| Lists consume more memory | Tuple consume less memory as compared to the list |
| Lists have several built-in methods | Tuple does not have many built-in methods. |
| The unexpected changes and errors are more likely to occur | In tuple, it is hard to take place. |

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

abc=(42)

print(abc)

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

List values from tuple

abc=(42,[1,2,3])

print(abc[1][1])

o/p 2

Tuple values from list

abc =(a,2, [1,2,3,4,5,6])

abc[2][1]

o/p 2

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. But for strings and integer values, variables simply contain the string or integer value. Python uses references whenever variables must store values of mutable data types, such as lists or dictionaries

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

copy module provides these two functions. When you use assignment operator Python just copies the references, not whole copy of the object.

copy performs shallow copy while deepcopy performs deep copy

import copy

aggv = [1, 2, ['a','c']]

c = copy.copy(aggv)

c[1] =5

c[2][1]= 10

print(c)

print(aggv)

o/p

[1, 5, ['a', 10]]

[1, 2, ['a', 10]]

import copy

aggv = [1, 2, ['a','c']]

c = copy.deepcopy(aggv)

c[1] =5

c[2][1]= 10

print(c)

print(aggv)

o/p

[1, 5, ['a', 10]]

[1, 2, ['a', 'c']]