1. What exactly is []?

Answer: [] , square brackets are used to specify list type of data.

[] denotes an empty list.Example,

lst=[] # here and empty list is prepared with variable name lst.

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.)

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

Answer: This task can be done by indexing operation, as 'hello' is the value to be assigned at position 3, its respective index will be 2. It can be written as ;

spam[2]= 'hello' # assigning 'hello' value to index number 2

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

Answer:

1. **'3' * 2** will give 33 in string datatype as '3' is string.
2. **'33'** is converted into integer using int() function.
3. **33/11** this division value gives float output 3.0
4. Finally 3.0 is converted into 3 using int().
5. Spam[3] gives value 'd' which is at index 3.
6.

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

Answer: Value of spam[-1] is d.

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

Let's pretend bacon has the list [3.14, 'cat,' 11, 'cat,' True] for the next three questions.

Answer : Value of spam[:2] is ['a','b']

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

Answer: 'cat' is at index 1 of bacon list, so value of bacon.index('cat') is 1.

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

Answer: append() method adds the value at end index of the existing list so bacon list will be,

[3.14, 'cat,' 11, 'cat,' True,99].

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

Answer: remove() removes first instance of the provided value from the list. bacon list now be,

[3.14, 11, 'cat,' True,99].

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

Answer: List concatenation operator is + . List Replication operators is *.

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

Answer: append() method adds value at the end of the list. Example ,

list1=[1,2,3,4]

list1.append(5)

print(list1)

Output: [1,2,3,4,5]

insert() method adds value at the provided index of the list.

list2=[1,2,4,5]

list2.insert(2,3) #value 3 is inserted at index 2.

print(list2)

Output : [1,2,3,4,5]

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

Answer: remove() method removes from the list ,first instance of value provided in the remove() method.

pop() method removes value at the index from the list , mentioned in the pop() method.

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

Answer :

1. List and string are similar in slicing and indexing operation.
2. List and string can be used with len() function.
3. For looping is identical in list and string.

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

Answer : Tuples are defined within (), while lists are defined within [].

List are mutable so slicing, indexing operations are possible in list, but on other hand tuples are immutable.

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

Answer: When only single value is present in tuple, the value must be followed with a comma in ().

Example:

Tuple_num=(42,)

If it is just written as (42), type won't be tuple will be integer.

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

Answer: list( ) function converts a tuple into list . tuple() function coverts list in tuple form.

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

Answer: They contain addresses referencing to the list values.

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

Answer : When an object is copied using copy(), it is called shallow copy as changes made in copied object will also make corresponding changes in original object, because both the objects will be referencing same address location.

When an object is copied using deepcopy(), it is called deep copy as changes made in copied object will not make corresponding changes in original object, because both the objects will not be referencing same address location.