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

**Ans:** These squared brackets are used to represent empty list.It means there is no item in the 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.)**

**Ans:**

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

spam [3] = “hello”

print(spam)

**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)]?**

**Ans:**  Output will be ‘d’

{solution: 33/11= 3, so look for 3rd index}

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

**Ans:** Output will be ‘d’ as negative indexing starts from end.

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

**Ans:** Output will be [‘a’,‘b’] as program will print till 1st index. Because 2(ending index)-1 = 1.

**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')?**

**Ans:**

* Output will be **invalid syntax** because commas (,) are inside the single quotes. It is not a proper list. It should be **[3.14, 'cat', 11, 'cat' ,True]**
* If the list is correct than output will be **1**. As list will not show the index of supplicate values

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

**Ans:** If the correct list is bacon = [3.14, 'cat', 11, 'cat' ,True]

Then output will be :

**bacon = [3.14, 'cat', 11, 'cat', True, 99]**

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

**Ans: Output will be** bacon = [3.14, 'cat', 11, True, 99]

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

**Ans:** list concatenation operator is **+** and list replication operator is **\***

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

**Ans:** Append () : It adds to the end of list. It takes only one argument and We can’t use index value to add any element at any particular place.

Insert (): To add the elements by their index value we use insert method.

**list.insert(index, element)**

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

**Ans :** remove method **------ list.remove(element)**

Pop method **------ list.pop(index)**

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

**Ans :** A **string is a sequence** of characters between single or double quotes. A **list is a sequence** of items, where each item could be anything (an integer, a float, a string, etc).

Both strings and lists have lengths: a string's length is the number of characters in the string; a list's length is the number of items in the list.

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

**Ans: List Tuple**

|  |  |
| --- | --- |
| 1. Represented by square brackets [] | **1.**Represented by round brackets () |
| 1. Lists are mutable | 2.Tuples are immutable |
| 1. Sort () is allowed | 3. sort () is not allowed |
| 1. Editing is allowed. We can insert and remove elements from the list easily | 4. But in Tuple, insert and remove are not allowed. It has some protective behaviour. |

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

**Ans:** tuple = (42)

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

**Ans:**

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

A = (1,2,3,4)

B = list(A)

Print(B, type(B))

**Output = [1,2,3,4] , ‘list’**

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

C = [1,2,” Python”,3,4]

D = (\*C,)

Print(D)

Output = (1,2,” Python”,3,4)

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

**Ans:** They could be references to list values or they can simply contain string and integer value.

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

**Ans : copy.copy()** creates references to original object. If you change copied object than original object will be changed. But

**Deepcopy()** creates a new object and it does real copying of original object to new one. Because of this original object will not be affected.