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

**The symbol "[]" represents an empty list in Python. A list is a data structure that can hold a collection of items or elements. When you use "[]" with nothing inside it, it signifies that the list is empty, meaning it contains no elements.**

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.)**

**Spam=[2,4,6,8,10]**

**Spam[2]=’Hello’**

**Print(spam)**

**Ans: [2,4,’Hello’,8,10]**

**Spam=[2,4,6,8,10]**

**Spam.insert(2,’Hello’)**

**Print(spam)**

**Ans : [2,4,’Hello’,6,8,10]**

**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)]?**

**‘d’**

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

‘**d’**

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

**[‘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')?**

**TypeError: insert expected 2 arguments, got 1**

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

**[3.14,’cat’,11,’cat’,True,99]**

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

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

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

**List Concatenation (+ operator):**

**L1=[1,2,3]**

**L2=[3,4,5]**

**L1 + L2 ----[1,2,3,4,5,6]**

list1 l1L **List Replication: L1=[1,2,3]**

**L1\*3 ---[1,2,3,1,2,3,1,2,3]**

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

**Append(): Add the element at the end of the list.list grows by size 1.**

**Insert(): Insert the element at the specific index pushing the elenernt in that index therefore increase the size of the list.**

**Spam=[2,4,6,8,10]**

**Spam.append(2)**

**Print(spam)**

**Ans: [2,4,8,10,2]**

**Spam=[2,4,6,8,10]**

**Spam.insert(2,’Hello’)**

**Print(spam)**

**Ans : [2,4,’Hello’,6,8,10]**

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

**Remove : remove first occurance of a specific value from a list.**

**L1=[1,2,3,2,4]**

**L1.remove(2)**

**Ans: [1,3,2,4]**

**Pop : this method is used to remove and return an item at an specific index in a list.**

**L1=[1,2,3,4]**

**L2=L1.pop(1)**

**#Return [1,3,4] #poped item is 2 which is in index 1**

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

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

**Lists: Lists are mutable, which means we can change their contents (add, remove, or modify elements) after they are created.**

**Tuples: Tuples are immutable, meaning once you create a tuple, we cannot change its contents. we cannot add, remove, or modify elements in a tuple.**

**Lists: Lists are slightly less memory-efficient and performant compared to tuples because of their mutability. When we modify a list, it may require resizing and copying elements to accommodate changes.**

**Tuples: Tuples are more memory-efficient and generally faster than lists when we don't need to change the elements because they are stored in a more compact form.**

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

**Tuple1=(42,)**

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

**My\_list=[1,2,3]**

**My\_tuple=tuple(My\_list)**

**Ans: (1,2,3)**

**Tuple1=(1,2,3,4)**

**List1=list(Tuple1)**

**Ans : [1,2,3,4]**

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

**When we create a variable and assign it a list, the variable stores a reference to the memory location where the list is stored. This reference essentially points to the list's data in memory.**

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

**copy.copy() creates a shallow copy, which shares references to inner objects, while copy.deepcopy() creates a deep copy, which duplicates all objects within the original object, resulting in no shared references between the original and copied objects.**