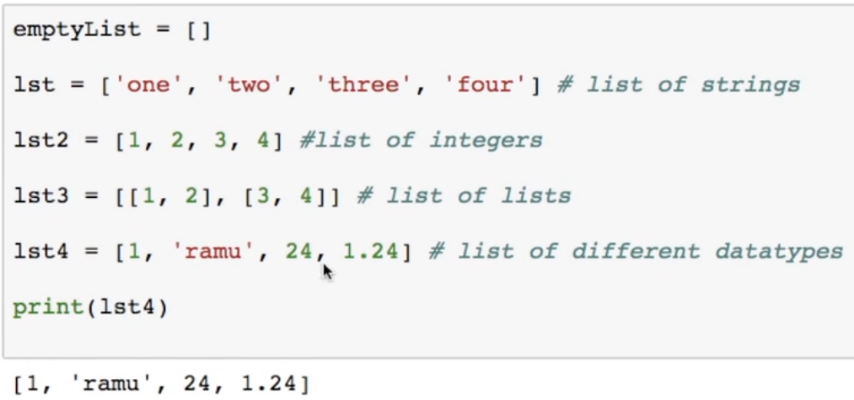
Lists

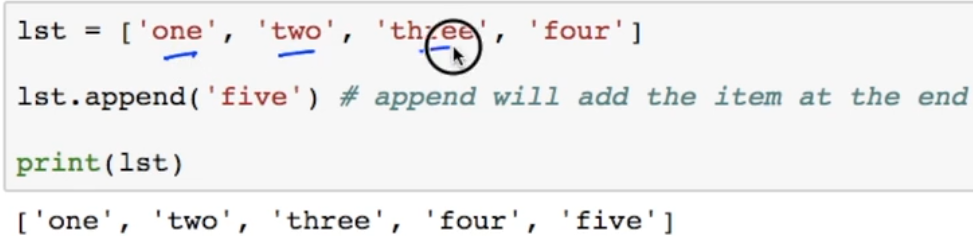
Data Structures:

A data structure is a collection of data elements such as numbers or characters or even other data structures that is structured in some way for example by numbering the elements. The most basic data structure in Python is the “sequence”.

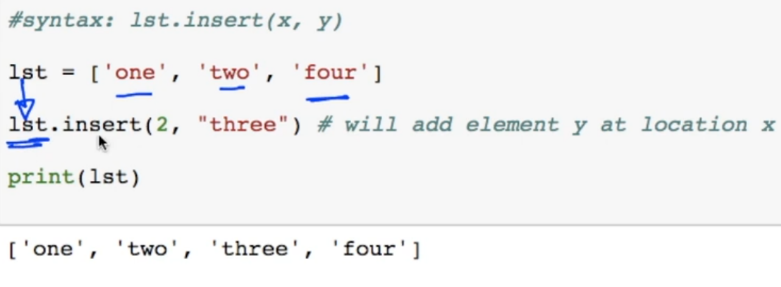
* List is one of the Sequence data structures.
* Lists are collection of items(Strings,integers or even other lists).
* Lists are enclosed in [].
* Each item in the list has an assigned index.
* Each item in the list is separated by commna.
* *Lists are mutable, which means they can be changed*.
* List Operations
* **Create**

****

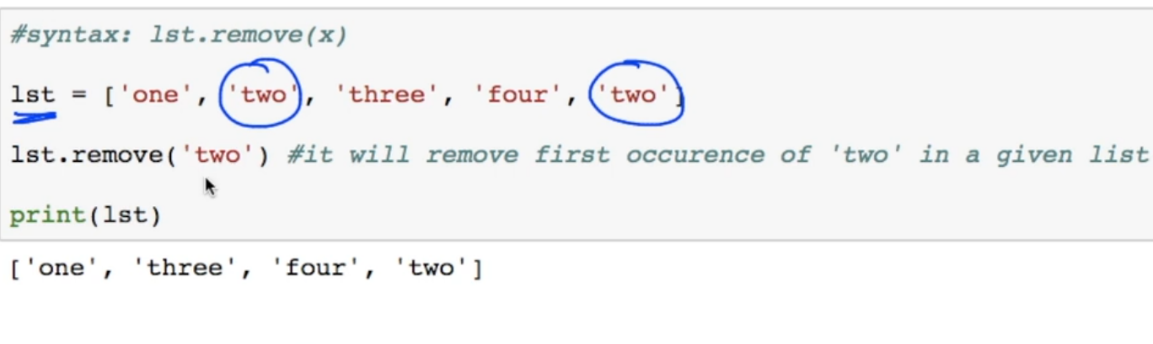
* **Append** 🡪 This will add the item at the end



* Insert 🡪



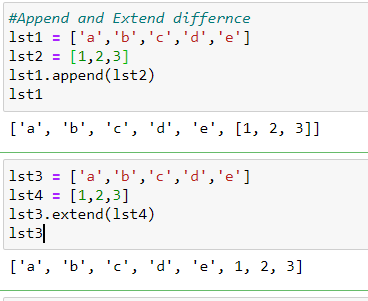
* Remove
  + This function removes the first occurrence of the passed element



* Append and Extend

This is will extend the list , it will not append it as different list.

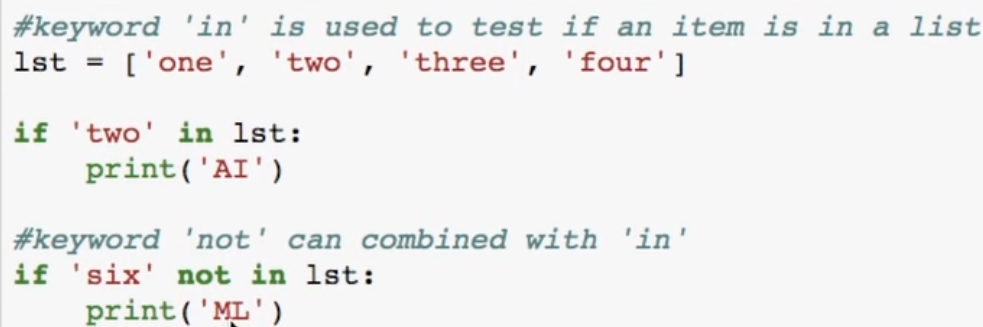
It will add the elements of list2 will be added to list1 as elements.

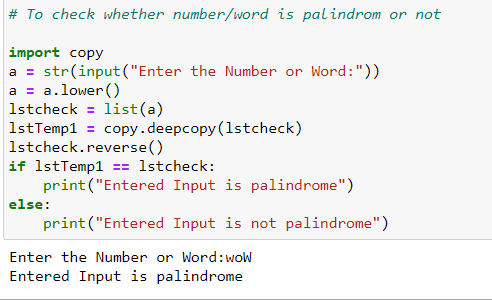


* Delete and Pop

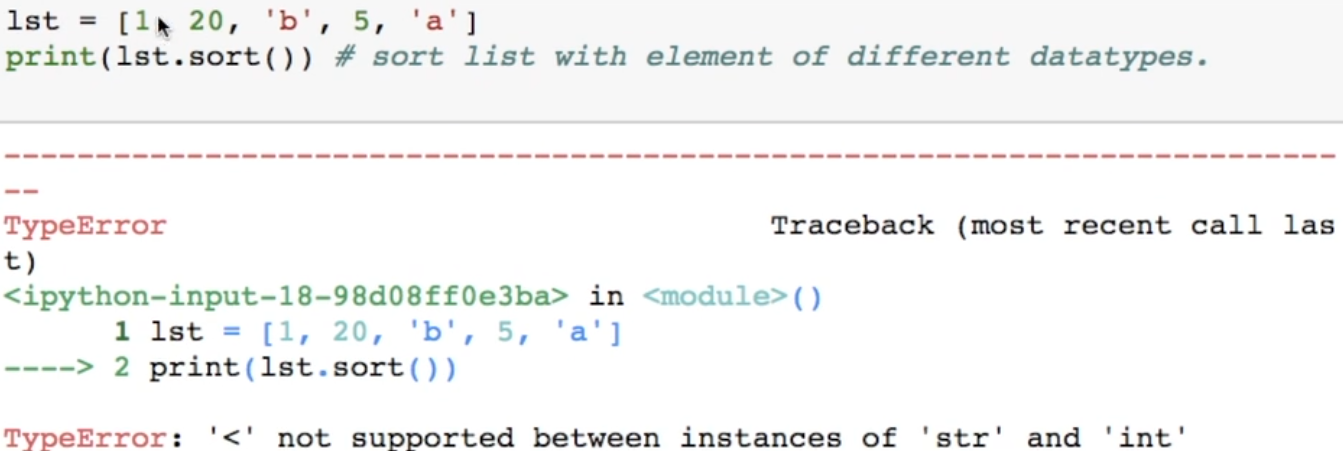
Delete deletes the item from list, pop just pops out the item from list.

Pop(index),del lst[index] ,remove(item)

* List Related Keywords
* List Reverse

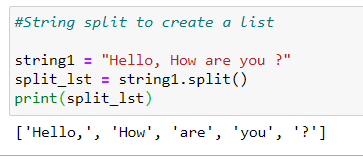


* List Sorting
  + Sorted(list, reverse=True/False) function returns a new list which is sorted, it does not modifies original list.
  + To modify the original list use sort() function.
  + If sort() is applied with list which is having different datatype then it throws an error.

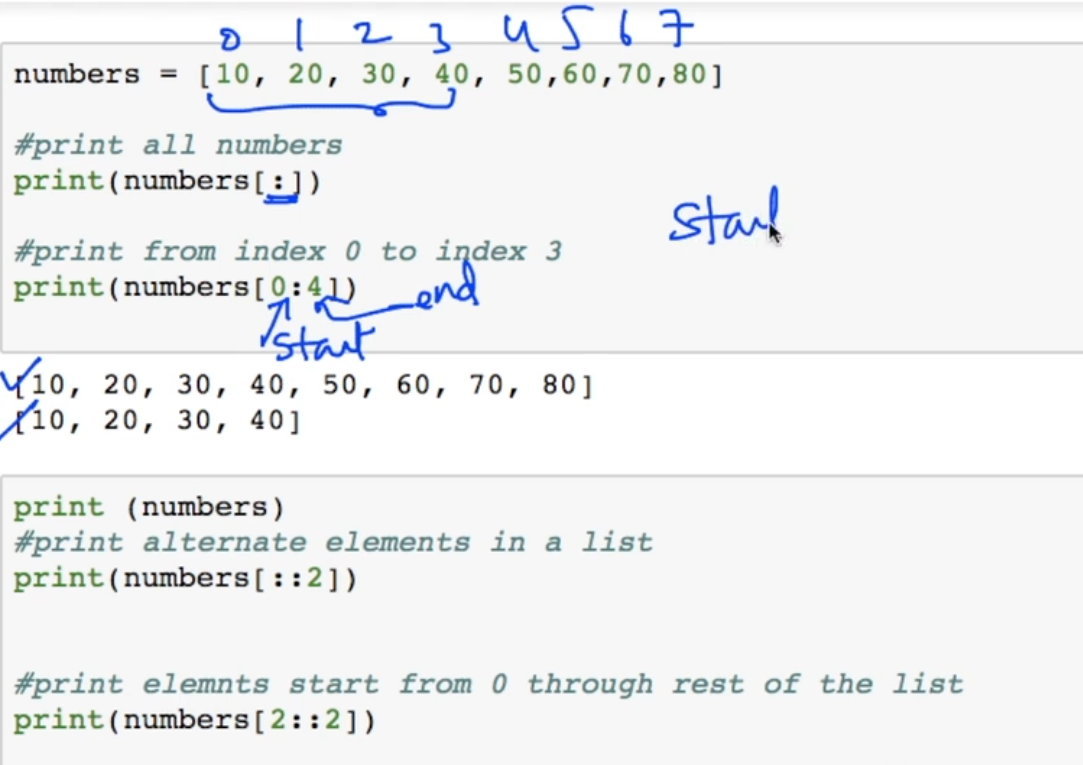


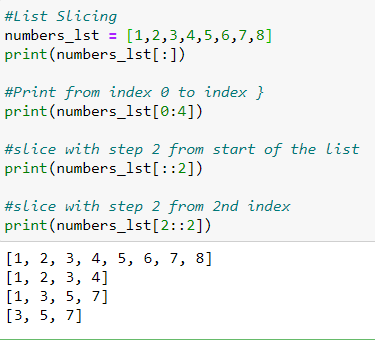
* String Split to create list



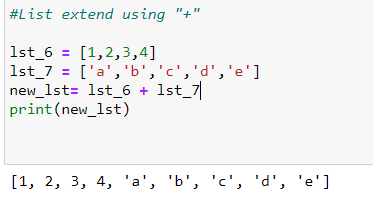


* List Indexing
  + List indexing can be done in both ways using positive or negative indexing.
  + Positive indexing gives values from the start of the list.
  + Negative indexing gives values from end of the list
* List Slicing
* List[start:end:stepsize]

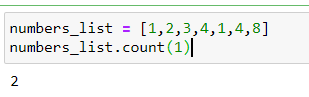




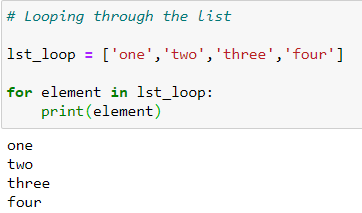
* List Extend using +



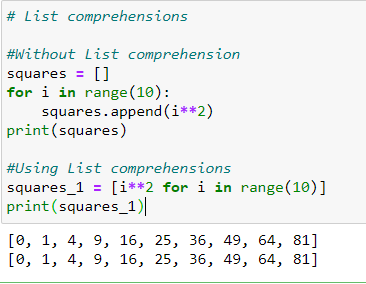
* List Count
  + List\_1.count(item)
  + It gives the count of frequency of an item in the list

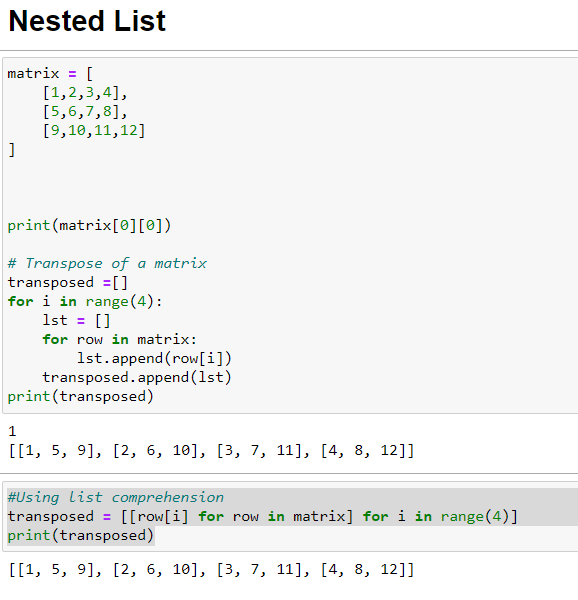


* Looping through the list



* List Comprehension
  + List comprehensions provide a concise way to create lists.
  + Common applications are to make new lists where each element is the result of some operation applied to each member of another sequence or iterable, or to create a subsequence of those elements that satisfy a certain condition.



* + *Nested List Comprehensions* **