Lists are basically a collection of ordered and mutable elements. In python lists are treated as arrays. Lists are identified with [ ] brackets. Let's create a simple list containing 5 items:

```
lst = [10,6,50,56,77]
print(lst)
```

Let's see some of list's different examples:

```
# list of strings
lst2 = ["brian","dom","mia","tez"]
print(lst2)

# list of dis-similar items
lst3 = ["brian",30,"mia",21,True,56.7,None]
print(lst3)
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 1_simple_list_creation.py
[10, 6, 50, 56, 77]
['brian', 'dom', 'mia', 'tez']
['brian', 30, 'mia', 21, True, 56.7, None]
```

Topic (insert new element into list): we can add new element into list using **.append()** method or using (+) operator. Note: whenever you add new elements externally it will add at the end of the list. Let's see the code:

```
# using append() method
lst = [10,60,70]
print(lst)
lst.append(600)
print(lst)

# using + operator
lst2 = ["brian","dom","mia"]
print(lst2)
lst2 = lst2 + ["letty"]
print(lst2)
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 2_append_in_lists.py
[10, 60, 70]
[10, 60, 70, 600]
['brian', 'dom', 'mia']
['brian', 'dom', 'mia', 'letty']
```

Topic (remove existing element from list): we can remove existing element from list using **.remove()** method. We can also use '**del'** keyword also or even using **pop()** method which we familiar with stack data structure. Note: when we applying pop() method we will remove last index element from the list.

```
# remove items using append() method
lst = [10,60,55,-90,400]
print(lst)
lst.remove(55)
print(lst)

# remove items using pop() method
lst2 = ["tez","cypher","dante"]
print(lst2)
lst2.pop()
print(lst2)

# remove element using 'del' keyword
lst3 = [12.4,34.5,67.8,90.89]
print(lst3)
del lst3[1]
print(lst3)
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 3_removing_items.py
[10, 60, 55, -90, 400]
[10, 60, -90, 400]
['tez', 'cypher', 'dante']
['tez', 'cypher']
[12.4, 34.5, 67.8, 90.89]
[12.4, 67.8, 90.89]
```

Let's see some of the list methods and their usages:

1. **len()** - we can get the length of list using len() method.
2. **type()** - it helps to get the datatype of the list.

```
lst = [10,50,"mia","tez"]
print(lst)
print(len(lst))
print(type(lst))

# adding or removing item's of list
lst.append(45.6)
print(lst," length now:",len(lst))
lst.remove("mia")
print(lst," length now: ",len(lst))
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 4_len_datatype.py
[10, 50, 'mia', 'tez']
4
<class 'list'>
[10, 50, 'mia', 'tez', 45.6]  length now: 5
[10, 50, 'tez', 45.6]  length now:  4
```

3. **insert()** – we can add new element at specified index of the list. When we use append() method it basically insert element at the end.
4. **index()** – this method helps to get element's index in the list.

```
lst = [10,50,"mia","tez","domnick"]
print("list is => ",lst)
for x in lst:
   print("item: ",x,"index: ",lst.index(x))

# let's add element "tez" into index 1
lst.insert(1,"tez")
print("list now => ",lst)
for x in lst:
   print("item: ",x,"index: ",lst.index(x))
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 5_insert_at_index.py
list is =>  [10, 50, 'mia', 'tez', 'domnick']
item:  10 index:  0
item:  50 index:  1
item:  mia index:  2
item:  tez index:  3
item:  domnick index:  4
list now =>  [10, 'tez', 50, 'mia', 'tez', 'domnick']
item:  10 index:  0
item:  tez index:  1
item:  50 index:  2
item:  mia index:  3
item:  tez index:  1
item:  domnick index:  5
```

5. **sort()** - we can sort list items using sort() method, which can sort list's items in ascending order.
6. **reverse()** – this method helps to reverse the whole list.

```
# let's sort list in ascending order
lst = [-70,67,33,12,34,3]
print(lst)
lst.sort()
print(lst)
```

```
# reverse list
lst2 = ["mia","brian","tez","domnick","letty"]
print(lst2)
lst2.reverse()
print(lst2)
```

```
C:\Users\shimanta das\Desktop\learn python\lists>python 6_sort_reverse.py
[-70, 67, 33, 12, 34, 3]
[-70, 3, 12, 33, 34, 67]
['mia', 'brian', 'tez', 'domnick', 'letty']
['letty', 'domnick', 'tez', 'brian', 'mia']
```