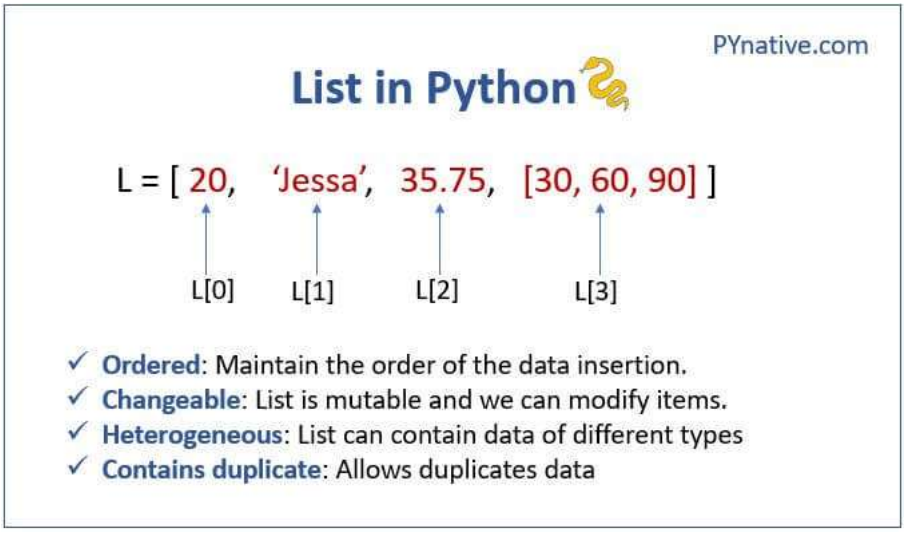
Python Lists



In [2]:
```
list=['a','b','c','d','e','f','g','h','i']
list
```

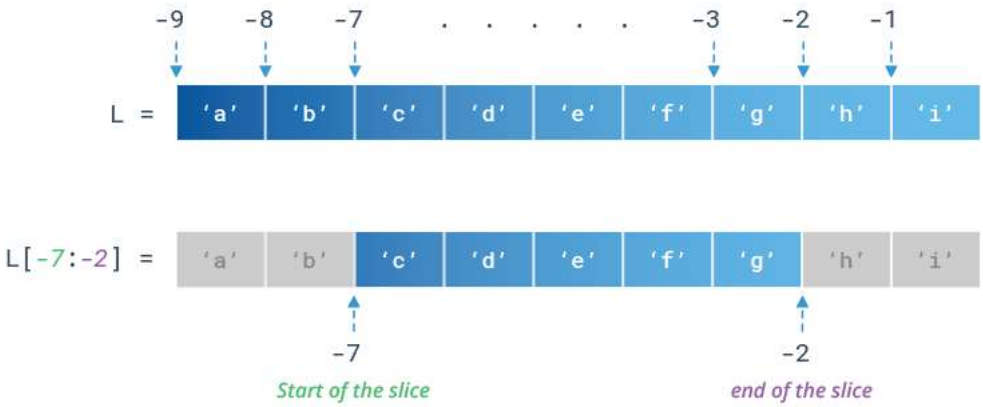
Out[2]:
```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
```

In [5]:
```
list[2:7]
```

Out[5]:
```
['c', 'd', 'e', 'f', 'g']
```

In [7]:
```
list[-7:-2]
```

Out[7]:
```
['c', 'd', 'e', 'f', 'g']
```



In [8]:
```
list[2:7:2]
```

Out[8]:
```
['c', 'e', 'g']
```

In [9]:
```
list[6:1:-2]
```

Out[9]:
```
['g', 'e', 'c']
```

You can reverse a list by omitting both start and stop indices and specifying a step as -1.

In [10]:
```
list[::-1]
```

Out[10]:
```
['i', 'h', 'g', 'f', 'e', 'd', 'c', 'b', 'a']
```

Omitting the start index starts the slice from the index 0. Meaning, L[:stop] is equivalent to L[0:stop]

In [11]:
```
list[:3]
```

Out[11]:
```
['a', 'b', 'c']
```

Whereas, omitting the stop index extends the slice to the end of the list. Meaning, L[start:] is equivalent to L[start:len(L)]

In [13]:
```
list[6:]
```

Out[13]:
```
['g', 'h', 'i']
```

Modify Multiple List Values

You can modify multiple list items at once with slice assignment. This assignment replaces the specified slice of a list with the items of assigned iterable.

In [14]:
```
list=['a','b','c','d','e','f','g','h','i']
list[1:3]=[1,2]
list
```

Out[14]:
```
['a', 1, 2, 'd', 'e', 'f', 'g', 'h', 'i']
```

In [15]:
```
list[1:2] = [1, 2, 3]
```

In [16]:
```
list
```

Out[16]:
```
['a', 1, 2, 3, 2, 'd', 'e', 'f', 'g', 'h', 'i']
```

Insert Multiple List Items

You can insert items into a list without replacing anything. Simply specify a zero-length slice.

In [17]:
```
list[:0]=['abc',32,45.4]
```

In [18]:
```
list
```

Out[18]:
```
['abc', 32, 45.4, 'a', 1, 2, 3, 2, 'd', 'e', 'f', 'g', 'h', 'i']
```

In [19]:
```
list[len(list):]=['yippie']
```

In [20]:
```
list
```

Out[20]:
```
['abc', 32, 45.4, 'a', 1, 2, 3, 2, 'd', 'e', 'f', 'g', 'h', 'i', 'yippie']
```

You can insert items into the middle of list by keeping both the start and stop indices of the slice same.

In [21]:
```
list[3:3]=["Denmark"]
```

In [22]:
```
list
```

Out[22]:
```
['abc',
 32,
 45.4,
 'Denmark',
 'a',
 1,
 2,
 3,
 2,
 'd',
 'e',
 'f',
 'g',
 'h',
 'i',
 'yippie']
```

Delete Multiple List Items

You can delete multiple items out of the middle of a list by assigning the appropriate slice to an empty list.

In [23]:
```
list[2:5]=[]
```

In [24]:
```
    list
```

Out[24]:
```
['abc', 32, 1, 2, 3, 2, 'd', 'e', 'f', 'g', 'h', 'i', 'yippie']
```

Clone or Copy a List

When you execute new_List = old_List, you don't actually have two lists. The assignment just copies the reference to the list, not the actual list. So, both new_List and old_List refer to the same list after the assignment.

You can use slicing operator to actually copy the list (also known as a shallow copy).

In [25]:
```
list1=list[:]
```

In [26]:
```
list1
```

Out[26]:
```
['abc', 32, 1, 2, 3, 2, 'd', 'e', 'f', 'g', 'h', 'i', 'yippie']
```

In [28]:
```
print(list1 is list )
```

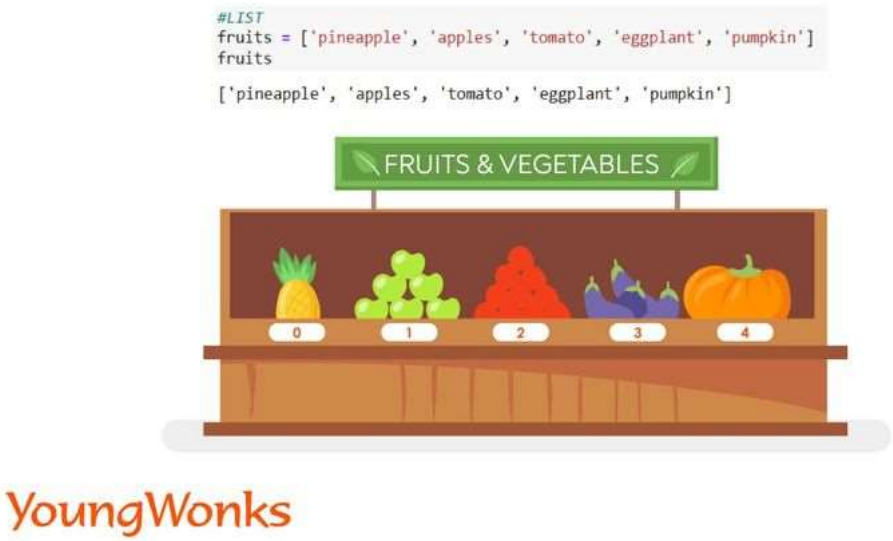
```
False
```

In [29]:
```
list==list1
```

Out[29]:
```
True
```

LIST METHODS USING FRUIT BASKETS

1.Create the list of Fruits



In [100]:
```
fruits_v=['pineapple','apples','tomato','eggplant','pumpkin']
fruits_v
```

Out[100]:
```
['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
```

2. Count number of items in a list

Syntax: len(list)

In [101]:
```
len(fruits_v)
```

Out[101]:
```
5
```

3. Add Items into a list

We can add items into a list using two ways.

Using APPEND AND INDEX

In [102]:
```
fruits_v.append("watermelon") #Using the name of the item
```

In [103]:
```
fruits_v
```

Out[103]:
```
['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin', 'watermelon']
```

```
# ADD ITEM USING NAME OF THE ITEM
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits.append('watermelon')
fruits
```

```
['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin', 'watermelon']
```

YoungWonks

In [104]:
```
fruits_v.insert(2,"sapodilla")
```

In [105]:
```
fruits_v
```

Out[105]:
```
['pineapple',
 'apples',
 'sapodilla',
 'tomato',
 'eggplant',
 'pumpkin',
 'watermelon']
```

4. Remove Items from a list

We can remove items from a list using two ways.

Using the NAME OF THE ITEM and using INDEX

In [106]:
```
fruits_v
```

Out[106]:
```
['pineapple',
 'apples',
 'sapodilla',
 'tomato',
 'eggplant',
 'pumpkin',
 'watermelon']
```

In [107]:
```
fruits_v.remove("tomato")
```

In [108]:
```
fruits_v
```

Out[108]:
```
['pineapple', 'apples', 'sapodilla', 'eggplant', 'pumpkin', 'watermelon']
```

```
# REMOVE ITEM USING NAME OF THE ITEM
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits.remove('tomato')
fruits
```

```
['pineapple', 'apples', 'eggplant', 'pumpkin']
```

YoungWonks

In [109]:
```
fruits_v.pop(1)
```

Out[109]:
```
'apples'
```

In [110]:
```
fruits_v
```

Out[110]:
```
['pineapple', 'sapodilla', 'eggplant', 'pumpkin', 'watermelon']
```

The default value is -1. -1 is the index of the last item there it removes the last item from the list.

This method has a return value. Similarly, to remove the first item you can specify the index as 0.

```
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits.pop(2)
fruits
```

```
['pineapple', 'apples', 'eggplant', 'pumpkin']
```

YoungWonks

5. Get items from a list

We can use index to get list items.

Syntax: list[index of the element]

In [111]:
```
fruits_v[3]
```

Out[111]:
```
'pumpkin'
```

6. Find position of an Item

This method returns the index of the element.

```
#GET ANY ITEM OF THE LIST
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits[1]
```

```
'apples'
```

```
# FIND POSITION OF AN ITEM
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
item = fruits.index('apples')
item
```

```
1
```

In [52]:
```
fruits_veggies=['pineapple','apples','tomato','eggplant','pumpkin']
fruits_veggies
```

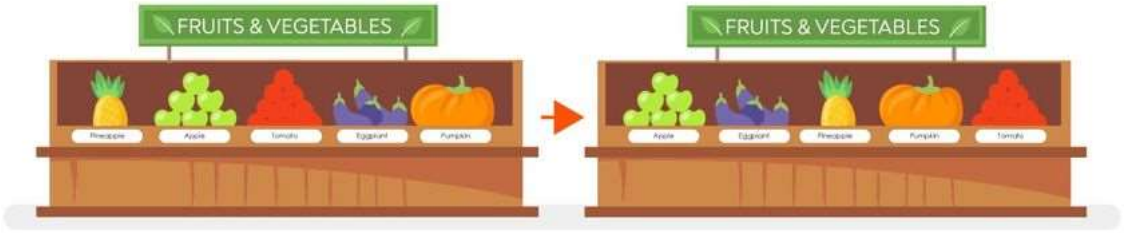
Out[52]:
```
['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
```

7. Sort the list

Using this method, the sorted list returns strings into alphabetical order and a list of numbers in ascending order.

```
# SORT LIST
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits.sort()
fruits
```

```
['apples', 'eggplant', 'pineapple', 'pumpkin', 'tomato']
```



YoungWonks

In [53]:
```
fruits_veggies.sort()
```

In [54]:
```
fruits_veggies
```

Out[54]:
```
['apples', 'eggplant', 'pineapple', 'pumpkin', 'tomato']
```

In [55]:
```
list_1=[1,24,5,78,56]
list_1
```

Out[55]:
```
[1, 24, 5, 78, 56]
```

In [56]:
```
list_1.sort()
```

In [57]:
```
list_1
```

Out[57]:
```
[1, 5, 24, 56, 78]
```

8. Reverse the list

In this method, the elements of the list is reversed.

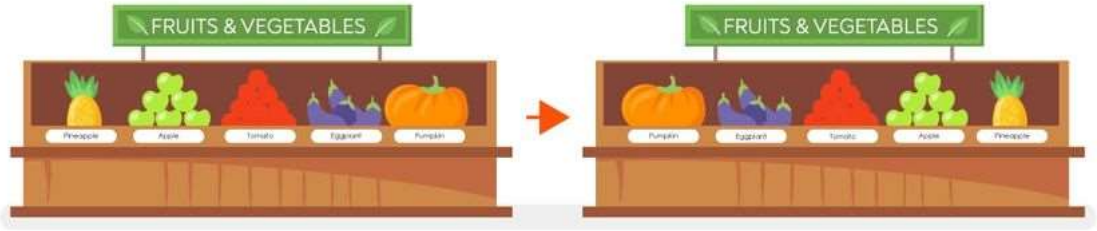
In [59]:
```
fruits_veggies.reverse()
```

In [60]:
```
fruits_veggies
```

Out[60]:
```
['tomato', 'pumpkin', 'pineapple', 'eggplant', 'apples']
```

```
# REVERSE LIST
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits.reverse()
fruits
```

```
['pumpkin', 'eggplant', 'tomato', 'apples', 'pineapple']
```



YoungWonks

In [62]:
```
list_1.sort(reverse=1) #descending order
```

In [63]:
```
list_1
```

Out[63]:
```
[78, 56, 24, 5, 1]
```

9. Replace list items

We make use of index to replace the items of a list.

Syntax: list[index of the element] = new item

In [65]:
```
fruits_veggies1=['pineapple','apples','tomato','eggplant','pumpkin']
```

In [66]:
```
fruits_veggies1
```

Out[66]:
```
['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
```

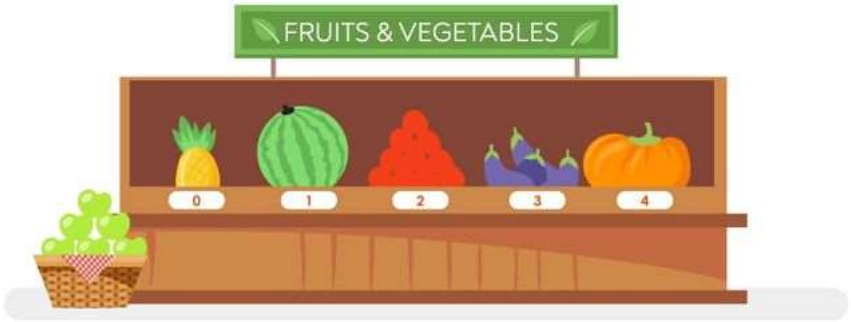
In [67]:
```
fruits_veggies1[1]='watermelon'
```

In [68]:
```
fruits_veggies1
```

Out[68]:
```
['pineapple', 'watermelon', 'tomato', 'eggplant', 'pumpkin']
```

```
# REPLACE ITEM
fruits = ['pineapple', 'apples', 'tomato', 'eggplant', 'pumpkin']
fruits[1] = 'watermelon'
fruits
```

```
['pineapple', 'watermelon', 'tomato', 'eggplant', 'pumpkin']
```



YoungWonks

10. Count the number of specific items

This method returns of the count of the item in the list.

Syntax: list.count(item name)

In [69]:
```
fv=['pineapple','apples','tomato','eggplant','pumpkin','apples','sapodilla','banana','banana','banana']
```

In [70]:
```
fv
```

Out[70]:
```
['pineapple',
 'apples',
 'tomato',
 'eggplant',
 'pumpkin',
 'apples',
 'sapodilla',
 'banana',
 'banana',
 'banana']
```

In [71]:
```
fv.count('banana')
```

Out[71]:
```
3
```

In [72]:
```
fv.count('apples')
```

Out[72]:
```
2
```

11. Make a copy of the list

Lists can be copied to create its duplicate and use it.

Syntax: new list = list.copy()

In [73]:
```
cities=['Bangalore','Mumbai','Delhi','Kolkata']
```

In [74]:
```
cities
```

Out[74]:
```
['Bangalore', 'Mumbai', 'Delhi', 'Kolkata']
```

In [77]:
```
cities1=cities.copy()
```

In [78]:
```
cities1
```

Out[78]:
```
['Bangalore', 'Mumbai', 'Delhi', 'Kolkata']
```

12. Add Multiple items using one-step

Here, we can add multiple items in the form of a list and the extend method is used to add all of the items into the original list.

Syntax: list.extend([item1, item2])

In [80]:
```
cities.extend(['Bhubaneshwar','Patna'])
```

In [81]:
```
cities
```

Out[81]:
```
['Bangalore', 'Mumbai', 'Delhi', 'Kolkata', 'Bhubaneshwar', 'Patna']
```

13. Empty the list

This function removes all the items of the list.

Syntax: list.clear()

In [82]:
```
cities.clear()
```

In [83]:
```
cities
```

Out[83]:
```
[]
```

14. Get each item of the list

Lists are iterable i.e. we can make use of a loop on it.

Syntax:

for i in list:

```
    print (i)
```

In [86]:
```
for i in fv:
    print(i)
```

```
pineapple
apples
tomato
eggplant
pumpkin
apples
sapodilla
banana
banana
banana
```

15. Maximum and Minimum number from the list.

Syntax: max(list); min(list)

This returns the maximum and minimum number from a list of numbers.

In [87]:
```
max(list_1)
```

Out[87]:
```
78
```

In [88]:
```
min(list_1)
```

Out[88]:
```
1
```

16. Random module on lists

We can use a few random module functions on a python list. This can perform specific actions on the list such as picking random item, jumble up the items of a list.

a. Pick random item from a list:

Syntax: random.choice(list)

b. Pick random item from a list as a list:

Syntax: random.sample(list, 1)

Here, the second parameter is to specify how many items we require. ¶

c. Randomly jumble the items of a list.

Syntax: random.shuffle(list)

In [89]:
```
import random
```

In [90]:
```
fv
```

Out[90]:
```
['pineapple',
 'apples',
 'tomato',
 'eggplant',
 'pumpkin',
 'apples',
 'sapodilla',
 'banana',
 'banana',
 'banana']
```

In [92]:
```
random.choice(fv)
```

Out[92]:
```
'pumpkin'
```

In [94]:
```
random.sample(fv,3)
```

Out[94]:
```
['tomato', 'pumpkin', 'sapodilla']
```

In [95]:
```
random.shuffle(fv)
```

In [96]:
```
fv
```

Out[96]:
```
['tomato',
 'banana',
 'sapodilla',
 'banana',
 'eggplant',
 'apples',
 'pumpkin',
 'apples',
 'banana',
 'pineapple']
```