DATA STRUCTURE IN PYTHON

Data Stucture ------> is the user or developer assign more than one(1) value in the data stucture....

Data structure is Divived into 2(two) parts

1. Inbuild data structure / Builtin Data Structure

(i) - LIST

(ii) - TOUPLE

(iii) - SET

(iv) - DICTONARY

(v) - RANGE

.This are the inbuild Data Stucture.

2. User Define Data Structure

(i) - STACK

(ii) - HEAP

(iii) - LINKEDLIST

(iv) - TREE

(v) - QUEUE

(vi) - HASHMAP

1 - LIST []

(i) - List is Define with Square Brackets []

(ii) - We can store stored Set of Values in the LIST.

(iii) - It can store elements of different types

(integer,float,string,bool,complex).

(iv) - "List are MUTABLE (Changeble) in python"

L = [] # l is variable and [] square barackets is define a list.. L

In [9]:

```
L = [1] # l is variable and in the [] square baracket one(1) is assign value.
L
```

```
Out[9]:  [1]
```

```
In [10]:  L = [3.14] # float value is allowed in list.
          L
```

```
Out[10]:  [3.14]
```

```
In [11]:  L = ['razekh'] # string value is allowed in list.
          L
```

```
Out[11]:  ['razekh']
```

```
In [12]:  L = [1 + 2j] # complex value is allowed in list.
          L
```

```
Out[12]:  [(1+2j)]
```

```
In [13]:  L = [True] # bool value is allowed in list.
          L

          # Over all in the list multiple Data type are allowed.
```

```
Out[13]:  [True]
```

```
In [14]:  type(L) # TYPE OF Data Stucture.
```

```
Out[14]:  list
```

```
In [ ]:  type(L) # TYPE OF Data Stucture.
```

```
In [15]:  id(L) # Address of variable L..
```

```
Out[15]:  2420894127744
```

```
In [16]:  L = ['shaikh razekh']
          L[0] = 'SHAIKH' # List is MUTABLE in python
          print(L)
```

```
['SHAIKH']
```

Methods Of LIST

Append()

Copy()

duplicate() , etc

Apepend

- Append means add the elements at the last...

```
In [18]:  L.append # - Append means add the elements at the last...
```

```
Out[18]:  <function list.append(object, /)>
```

```
In [17]:  L.append
```

```
Out[17]:  <function list.append(object, /)>
```

```
In [22]:  L.append(11) # 11 add at the last.
          L
```

```
Out[22]:  ['SHAIKH', 10, 11, 11, 11]
```

```
In [23]:  L.append(30) #First Argument (30)
          L.append(35) #First Argument (35)
          L.append(40) #First Argument (40)
          print(L)
```

```
['SHAIKH', 10, 11, 11, 11, 30, 35, 40]
```

```
In [24]:  L.append # add element at last
          L
          (30)
```

```
Out[24]:  30
```

```
In [26]:  len(L)
```

```
Out[26]:  8
```

```
In [27]:  L1 = L.copy()
          L1
```

```
Out[27]:  ['SHAIKH', 10, 11, 11, 11, 30, 35, 40]
```

```
In [28]:  L == L1 # l is equal to l1
```

```
Out[28]:  True
```

```
In [29]:  L != L1 # l is equal not to l1
```

```
Out[29]:  False
```

```
In [30]:  L != L1 # l is equal not to l1
```

```
Out[30]:  False
```

```
In [31]:  L1.append(100)
          L1
```

```
Out[31]:  ['SHAIKH', 10, 11, 11, 11, 30, 35, 40, 100]
```

```
In [32]:  L = L1
          L
```

```
Out[32]:  ['SHAIKH', 10, 11, 11, 11, 30, 35, 40, 100]
```

```
In [33]:  print(len(L)) #length of l
```

```
          print(len(L1)) #length of l

          9
          9

In [34]:  id(L1) # Address of l1

Out[34]:  2420894513984

In [35]:  len(L) # length of l

Out[35]:  9

In [36]:  L.clear() # All clear
          L

Out[36]:  []

In [37]:  len(L1)

Out[37]:  0

In [38]:  id(L1)

Out[38]:  2420894513984

In [41]:  L

Out[41]:  []

In [42]:  L1

Out[42]:  []

In [43]:  L.append(2.3,34)                            # pass only one argument not more than one.

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          Cell In[43], line 1
          ----> 1 L.append(2.3,34)

          TypeError: list.append() takes exactly one argument (2 given)

In [44]:  L1

Out[44]:  []

In [45]:  L.append('nit')
          L.append(3.6)
          L.append(1 + 2j)
          L.append([1,2,3])

In [46]:  L

Out[46]:  ['nit', 3.6, (1+2j), [1, 2, 3]]

In [47]:  L.append(True)
```

```
L
```

```
Out[47]: ['nit', 3.6, (1+2j), [1, 2, 3], True]
```

Copy

- Copy one list to Another List...

```
In [49]: s = ['Good Morning']
         s
```

```
Out[49]: ['Good Morning']
```

```
In [50]: s1 = s.copy() # copy method
         s1
```

```
Out[50]: ['Good Morning']
```

```
In [51]: s1 = {'Sunday''Monday'}
         print(s)
         print(s1)
```

```
['Good Morning']
{'SundayMonday'}
```

```
In [52]: print(id(s)) # Address of s
         print(id(s1))# Address of s
```

```
2420894522688
2420881420608
```

```
In [53]: print(len(s)) #length of s
         print(len(s1))#length of s
```

```
1
1
```

Join Lists

```
In [1]: list1 = ['one', 'two', 'three', 'four']
        list2 = ['five', 'six', 'seven', 'eight']
```

```
In [2]: list3 = list1 + list2 # Join two lists by '+' operator
        list3
```

```
Out[2]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [3]: list1.extend(list2) #Append list2 with list1
        list1
```

```
Out[3]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

List Membership

```
In [4]: list1
```

```
Out[4]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [5]: 'one' in list1 # Check if 'one' exist in the list
```

```
Out[5]: True
```

```
In [6]: 'ten' in list1 # Check if 'ten' exist in the list
```

```
Out[6]: False
```

```
In [ ]: if 'two' in list1: # Check if 'three' exist in the list
        print('Three is present in the list')
        else:
        print('Three is not present in the list')
```

```
In [ ]: if 'three' in list1: # Check if 'three' exist in the list
        print('Three is present in the list')
        else:
        print('Three is not present in the list')
```

```
In [ ]: if 'eleven' in list1: # Check if 'eleven' exist in the list
        print('eleven is present in the list')
        else:
        print('eleven is not present in the list')
```

Reverse & Sort List

```
In [12]: list1
```

```
Out[12]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [14]: list1.reverse() # Reverse the list
         list1
```

```
Out[14]: ['eight', 'seven', 'six', 'five', 'four', 'three', 'two', 'one']
```

```
In [15]: list1 = list1[::-1] # Reverse the list
         list1
```

```
Out[15]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [16]: mylist3 = [9,5,2,99,12,88,34]
         mylist3.sort() # Sort list in ascending order
         mylist3
```

```
Out[16]: [2, 5, 9, 12, 34, 88, 99]
```

```
In [17]: mylist3 = [9,5,2,99,12,88,34]
         mylist3.sort() # Sort list in ascending order
         mylist3
```

```
Out[17]: [2, 5, 9, 12, 34, 88, 99]
```

```
In [18]: mylist4 = [88,65,33,21,11,98]
         sorted(mylist4)

Out[18]: [11, 21, 33, 65, 88, 98]

In [19]: mylist4

Out[19]: [88, 65, 33, 21, 11, 98]
```

Loop through a list

```
In [20]: list1

Out[20]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']

In [ ]: for i in list1:
        print(i)

In [ ]: for i in enumerate(list1):
        print(i)
```

Count

```
In [27]: list10 =['one', 'two', 'three', 'four', 'one', 'one', 'two', 'three']

In [28]: list10.count('one')    # Number of times item "one" occurred in the list.

Out[28]: 3

In [29]: list10.count('two') # Occurence of item 'two' in the list

Out[29]: 2

In [30]: list10.count('four') #Occurence of item 'four' in the list

Out[30]: 1
```

All / Any

The all() method returns:

True - If all elements in a list are true False - If any element in a list is false

The any() function returns True if any element in the list is True. If not, any() returns False.

```
In [31]: L1 = [1,2,3,4,0]

In [32]: all(L1) # Will Return false as one value is false (Value 0)
```

```
Out[32]:  False

In [33]:  any(L1) # Will Return True as we have items in the list with True value

Out[33]:  True

In [34]:  L2 = [1,2,3,4,True,False]

In [35]:  all(L2) # Returns false as one value is false

Out[35]:  False

In [36]:  any(L2) # Will Return True as we have items in the list with True value

Out[36]:  True

In [37]:  L3 = [1,2,3,True]

In [38]:  all(L3) # Will return True as all items in the list are True

Out[38]:  True

In [ ]:
```