# DATA STRUCTURE

# LIST

# * list is a ordered data structure.

# * List is mutable i.e we can change or modify the items or data present in the list.

# * List can contain different datatypes like int,float,sring,complex,boolesn etc.

# * List has it's own diffenent attribute/function.Let's disscuss it.

# List Creation

In [1]:
```
List=[] # Empty list
```

In [2]:
```
print(type(list))
```

```
<class 'type'>
```

In [3]:
```
l1=[20,50,8,45,3,60] # List of integers
l1
```

Out[3]: [20, 50, 8, 45, 3, 60]

In [4]:
```
l2=[2.3,8.9,45.6,58.36] # List of float
l2
```

Out[4]: [2.3, 8.9, 45.6, 58.36]

In [5]:
```
l3=["hello","good","morning","Dear"] #List of strings
l3
```

Out[5]: ['hello', 'good', 'morning', 'Dear']

In [6]:
```
l4=[(2+3j),(5+6j),(45+4j)]# Ist of complex numbers
l4
```

Out[6]: [(2+3j), (5+6j), (45+4j)]

In [7]:
```
l5=[45,"hello",3.4,(43+9j),[3,4,5]]# List of mixed data types
l5
```

Out[7]: [45, 'hello', 3.4, (43+9j), [3, 4, 5]]

In [9]:
```
l6=[[12,43,56,78],["hello","oj",98,9.4]]#Nested list
```

# List Functions/Attribute

In [ ]:
```
l5
```

In [ ]:
```
len((l5))
```

In [ ]:
```
l5.index(3.4)
```

In [11]:
```
l5.append(85) #APPEND ADD ITEM ONLY AT THE LAST OF THE LIST
l5
```

Out[11]: [45, 'hello', 3.4, (43+9j), [3, 4, 5], 85]

In [ ]:
```
l5.append(45,80)
l5
```

In [ ]:
```
l5.clear()
l5
```

In [ ]:
```
l5
```

In [12]:
```
l5
```

Out[12]: [45, 'hello', 3.4, (43+9j), [3, 4, 5], 85]

In [14]:
```
l5.copy()
```

Out[14]: [45, 'hello', 3.4, (43+9j), [3, 4, 5], 85]

In [16]:
```
l5.count(85) #it counts the no of same items present in the list
```

Out[16]: 1

In [17]:
```
l5.append(85)
```

In [18]:
```
l5
```

Out[18]: [45, 'hello', 3.4, (43+9j), [3, 4, 5], 85, 85]

In [19]:
```
l5.count(85)
```

Out[19]: 2

In [20]:
```
l5.extend(l4) #it extend(add) the given list to the cuuent list
```

In [21]:
```
l5
```

Out[21]:
```
[45, 'hello', 3.4, (43+9j), [3, 4, 5], 85, 85, (2+3j), (5+6j), (45+4j)]
```

In [25]:
```
l5.insert(2,"python")
```

In [26]:
```
l5
```

Out[26]:
```
[45,
 'hello',
 'python',
 3.4,
 (43+9j),
 [3, 4, 5],
 85,
 85,
 (2+3j),
 (5+6j),
 (45+4j)]
```

In [28]:
```
l5.insert(3,6,"hi") #insert() only accept 2 argument
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[28], line 1
----> 1 l5.insert(3,6,"hi")

TypeError: insert expected 2 arguments, got 3
```

In [29]:
```
l5.pop(1) #it remove the element by the given index(indexwise)
```

Out[29]:
```
'hello'
```

In [30]:
```
l5
```

Out[30]:
```
[45, 'python', 3.4, (43+9j), [3, 4, 5], 85, 85, (2+3j), (5+6j), (45+4j)]
```

In [31]:
```
l5.remove(85)
l5
```

Out[31]:
```
[45, 'python', 3.4, (43+9j), [3, 4, 5], 85, (2+3j), (5+6j), (45+4j)]
```

In [32]:
```
l5.remove(85,85)  #it exactly takes one arg.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[32], line 1
----> 1 l5.remove(85,85)
      2 l5

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

In [34]:
```
l5.reverse()
```

In [35]:
```
l5
```

Out[35]: [(45+4j), (5+6j), (2+3j), 85, [3, 4, 5], (43+9j), 3.4, 'python', 45]

In [36]:
```
l5.sort() #sort() doesn't support betwwen different datatypes.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[36], line 1
----> 1 l5.sort()

TypeError: '<' not supported between instances of 'complex' and 'complex'
```

In [37]:
```
l7=[50,40,700,80000,96,53]
l7.sort()
```

In [38]:
```
l7
```

Out[38]: [40, 50, 53, 96, 700, 80000]

In [64]:
```
l8=[50,85,500,1,89,3]
l8.sort(reverse=True)
l8
```

Out[64]: [500, 89, 85, 50, 3, 1]

In [42]:
```
l7
```

Out[42]: [80000, 700, 96, 53, 50, 40]

In [53]:
```
l5
```

Out[53]: [(45+4j), (5+6j), (2+3j), 85, [3, 4, 5], (43+9j), 3.4, 'python', 45]

In [55]:
```
l5.extend([23,4,9]) # it adds more than one items to the exisiting list in list
l5
```

Out[55]: [(45+4j), (5+6j), (2+3j), 85, [3, 4, 5], (43+9j), 3.4, 'python', 45, 23, 4, 9]

# list operations

# Comparision operator

Lists with other lists (element-wise comparison in order),

List elements with values (inside loops or comprehensions).

In [14]:
```
list_1=[2,4,9,7]
list_2=[2,4,1,9]
list_3=[2,4,1,9]
```

In [15]:
```
list_1>list_2
```

Out[15]: True

In [16]:
```
list_1<list_2
```

Out[16]: False

In [17]:
```
list_3>list_1
```

Out[17]: False

In [18]:
```
list_3==list_1
```

Out[18]: False

In [20]:
```
list_1!=list_2
```

Out[20]: True

In [21]:
```
list_2==list_3
```

Out[21]: True

In [31]:
```
list10=["hell",2,2.3]
list11=[10,"ji",20]
```

In [32]:
```
list10==list11
```

Out[32]: False

In [33]:
```
list10<list11 # <,> operator not supported in list
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[33], line 1
----> 1 list10<list11

TypeError: '<' not supported between instances of 'str' and 'int'
```

In [ ]:

# all() and any() with comparison

In [25]:
```
list_1=[2,4,9,7]    # if all the numbers are greater than 0 then it prints true
print(all(x>0 for x in list_1))
```

True

In [26]:
```
list_1=[2,4,9,7]    # if all the numbers are less than 0 then it prints true
print(all(x<0 for x in list_1))
```

False

In [28]:
```
print(any(x < 10 for x in list_1)) # if any numbers are less than 10 then it pri
```

True

In [ ]:
```
list_1=[20,30,78,4,9]
list_2=[50,89,2,5,7]
```

```
list_1>list_2
```

In [ ]:

In [ ]:

# joining

In [39]:
```
print(l5+l7)
```

```
[(45+4j), (5+6j), (2+3j), 85, [3, 4, 5], (43+9j), 3.4, 'python', 45, 40, 50, 53, 96, 700, 80000]
```

In [43]:
```
list1=[4,9,59,"23",(5+6j),"hi"]
list2=[23,45,6,7,89,9]
list3=list1+list2
```

In [ ]:
```
list3
```

# del function

In [56]:
```
mylist=[1,34,5,67,5] #del function is used to delete entire list
del mylist
```

In [58]:
```
mylist
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[58], line 1
----> 1 mylist

NameError: name 'mylist' is not defined
```

In [59]:
```
del list3[1:3] # delete the item occupied to that index
list3
```

Out[59]:
```
[4, '23', (5+6j), 'hi', 23, 45, 6, 7, 89, 9]
```

In [62]:
```
del list3[9] # delete the particular int only from the list
list3
```

Out[62]:
```
[4, '23', (5+6j), 'hi', 23, 45, 6, 7, 89]
```

# list membership operator

In [69]:
```
mylist=["one",1,"two",2,"three",3]
"one"in mylist
```

Out[69]:
```
True
```

In [70]:
```
5 in mylist
```

Out[70]: False

In [72]:
```
"three" in mylist
```

Out[72]: True

# eval()

In [74]:
```
eval('5'+'6') #it evaluates the  int data present in str format
```

Out[74]: 56

In [78]:
```
eval('3-1')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[78], line 1
----> 1 eval('3'*'1')

TypeError: can't multiply sequence by non-int of type 'str'
```

In [80]:
```
eval('7'-'5') # eval function works for addition and for str format only
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[80], line 1
----> 1 eval('7'-'5')

TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

# max(),min(),len()

my_list=[56,800,9,45,7] mx=max(my_list) mx

In [83]:
```
mn=min(my_list)
mn
```

Out[83]: 7

In [85]:
```
len(my_list)
```

Out[85]: 5

In [ ]:
```
# replicate/repeat list
```

In [92]:
```
list2=[23,90,8]
list4=list2*3
list4
```

Out[92]: [23, 90, 8, 23, 90, 8, 23, 90, 8]

In [94]:
```
list2=[[1,2],[56,7],["hello","ji"]] #nested list replication
replica=list2*4
```

```
replica
```

Out[94]:
```
[[1, 2],
 [56, 7],
 ['hello', 'ji'],
 [1, 2],
 [56, 7],
 ['hello', 'ji'],
 [1, 2],
 [56, 7],
 ['hello', 'ji'],
 [1, 2],
 [56, 7],
 ['hello', 'ji']]
```

# list slicing(list[start:stop:step])

start: Index to begin from (inclusive)

stop: Index to stop before (exclusive)

step: Interval between elements (default is 1)

In [95]:
```
list4
```

Out[95]:
```
[23, 90, 8, 23, 90, 8, 23, 90, 8]
```

In [96]:
```
list4[:5]
```

Out[96]:
```
[23, 90, 8, 23, 90]
```

In [98]:
```
list4[1:8:2]
```

Out[98]:
```
[90, 23, 8, 90]
```

In [100…
```
list4[::3]
```

Out[100…
```
[23, 23, 23]
```

In [101…
```
list4[2::3]
```

Out[101…
```
[8, 8, 8]
```

In [103…
```
list4[0:8:]
```

Out[103…
```
[23, 90, 8, 23, 90, 8, 23, 90]
```

In [ ]:
```

```

# Loop in list

In [73]:
```
for i in mylist:
    print(i)
```

```
one
1
two
2
three
3
```

In [ ]:
```
# program to print element of a list['q','w','e','r','t','y'] in separate lines
```

In [36]:
```
L=['q','w','e','r','t','y']
length=len(L)
for i in range(length):
    print("At indexs",i,"and",(i-length),"element:",L[i])
```

```
At indexs 0 and element: q
At indexs 1 and element: w
At indexs 2 and element: e
At indexs 3 and element: r
At indexs 4 and element: t
At indexs 5 and element: y
```

In [ ]:
```

```