5th March

In [2]:
```
l=[10,30,40,2,5,2.3,34,55,2,4,1]
l
```

Out[2]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [3]:
```
l1=l.copy()
l1
```

Out[3]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [7]:
```
l2=[100,50,20]
l2
```

Out[7]: [100, 50, 20]

In [8]:
```
print(l)
print(l1)
print(l2)
```

```
[10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]
[10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]
[100, 50, 20]
```

String List slicing (Data type)

In [13]:
```
s = 'Ameerpet'
s
```

Out[13]: 'Ameerpet'

In [14]:
```
s[0]
```

Out[14]: 'A'

In [15]:
```
s[:]
```

Out[15]: 'Ameerpet'

In [16]:
```
s[:3]
```

Out[16]: 'Ame'

In [17]:
```
s[1:]
```

Out[17]: 'meerpet'

In [18]:
```
s[::-1]
```

Out[18]: 'tepreemA'

In [19]:
```
s
```

Out[19]: 'Ameerpet'

In [76]:
```
for i in enumerate(s):
    print(i)
```

```
(0, 'A')
(1, 'm')
(2, 'e')
(3, 'e')
(4, 'r')
(5, 'p')
(6, 'e')
(7, 't')
```

List Slicing

In [21]:
```
l
```

Out[21]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [22]:
```
l[:]
```

Out[22]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [23]:
```
l[1:]
```

Out[23]: [30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [24]:
```
l[:5]
```

Out[24]: [10, 30, 40, 2, 5]

In [26]:
```
l[0:7] # 2nd (n-1)
```

Out[26]: [10, 30, 40, 2, 5, 2.3, 34]

In [27]:
```
l[:-2]
```

Out[27]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2]

In [28]:
```
l
```

Out[28]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [29]:
```
l[0:100:5]
```

Out[29]: [10, 2.3, 1]

In [30]:
```
l
```

Out[30]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [31]:
```
l[2:10:2]
```

Out[31]: [40, 5, 34, 2]

In [32]:
```
len(l)
```

Out[32]: 11

In [33]:
```
l
```

Out[33]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [34]:
```
l[::-1]
```

Out[34]: [1, 4, 2, 55, 34, 2.3, 5, 2, 40, 30, 10]

In [36]:
```
l[::-2] # starts from last element
```

Out[36]: [1, 2, 34, 5, 40, 10]

In [37]:
```
l
```

Out[37]: [10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [38]:
```
l[0]
```

Out[38]: 10

In [41]:
```
l[0]=12 # List is Mutable or Hashable (changable)
l
```

Out[41]: [12, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [42]:
```
l[2]=20
l
```

Out[42]: [12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 1]

In [43]:
```
l[-1] = 'nit'
l
```

Out[43]: [12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']

In [51]:
```
l[-1][0] # Nested Slicing
```

Out[51]: 'n'

In [52]:
```
l[-1][1]
```

Out[52]: 'i'

In [53]:
```
l[-1][2]
```

Out[53]: 't'

In [56]:
```
print(l[-1])
```

```
nit
```

In [77]:
```
for i in enumerate(l[-1]):
    print(i)
```

```
(0, 'n')
(1, 'i')
(2, 't')
```

In [58]:
```
l
```

Out[58]: [12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']

In [60]:
```
print(l[-1][0])
print(l[-1][1])
print(l[-1][2])
```

```
n
i
t
```

In [61]:
```
l
```

Out[61]: [12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']

In [62]:
```
print(l)
print(l1)
print(l2)
print(l3)
```

```
[12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']
[10, 30, 40, 2, 5, 2.3, 34, 55, 2, 4, 1]
[100, 50, 20]
[100, 50, 20]
```

In [66]:
```
print(len(l))
print(len(l1))
print(len(l2))
print(len(l3))
```

```
11
11
3
3
```

In [67]:
```
l4 = l3+l2
```

In [68]:
```
l4
```

Out[68]:
```
[100, 50, 20, 100, 50, 20]
```

In [ ]:
```
# List Membership (checking whether a specific element exists)
```

In [69]:
```
10 in l2
```

Out[69]:
```
False
```

In [71]:
```
10 in l1
```

Out[71]:
```
True
```

Enumerate

In [ ]:
```
# a way to loop through a list (or any other collection) and get both the index (po
```

In [73]:
```
l
```

Out[73]:
```
[12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']
```

In [78]:
```
for i in enumerate(l):
    print(i)
```

```
(0, 12)
(1, 30)
(2, 20)
(3, 2)
(4, 5)
(5, 2.3)
(6, 34)
(7, 55)
(8, 2)
(9, 4)
(10, 'nit')
```

In [79]:
```
l
```

Out[79]:
```
[12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit']
```

In [86]:
```
all (l) # no zeroes in l (list)
```

Out[86]:
```
False
```

In [87]:
```
any (l)
```

Out[87]:
```
True
```

In [88]:
```
l.append(0)
l
```

Out[88]:
```
[12, 30, 20, 2, 5, 2.3, 34, 55, 2, 4, 'nit', 0, 0]
```

In [89]:
```
all (l) # zero exists in l
```

Out[89]:
```
False
```

In [90]:
```
any (l) # if a single non zero value exists the True
```

Out[90]:
```
True
```

In [ ]:

In [ ]:

In [ ]: