In [4]:
```
# Datastruture - user will define the value more then one - list, tuple, set, Dict
```

In [6]:
```
l=[]
l
```

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

In [8]:
```
len(l)
```

Out[8]:
```
0
```

In [10]:
```
l.append(10)
```

In [12]:
```
l
```

Out[12]:
```
[10]
```

In [16]:
```
len(l)
```

Out[16]:
```
1
```

In [18]:
```
l.append(20)
l.append(30)
l.append(40)
l.append(40)
```

In [20]:
```
l
```

Out[20]:
```
[10, 20, 30, 40, 40]
```

In [22]:
```
len(l)
```

Out[22]:
```
5
```

In [24]:
```
id(l)
```

Out[24]:
```
2282554396800
```

In [26]:
```
print(type(l))
```

```
<class 'list'>
```

In [28]:
```
import keyword
keyword.kwlist
```

Out[28]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 'break',
 'class',
 'continue',
 'def',
 'del',
 'elif',
 'else',
 'except',
 'finally',
 'for',
 'from',
 'global',
 'if',
 'import',
 'in',
 'is',
 'lambda',
 'nonlocal',
 'not',
 'or',
 'pass',
 'raise',
 'return',
 'try',
 'while',
 'with',
 'yield']
```

In [30]:
```
len(keyword.kwlist)
```

Out[30]:
```
35
```

In [32]:
```
l
```

Out[32]:
```
[10, 20, 30, 40, 40]
```

In [34]:
```
l[:]
```

Out[34]:
```
[10, 20, 30, 40, 40]
```

In [36]:
```
l[0]
```

Out[36]:
```
10
```

In [38]:
```
l[-1]
```

Out[38]: 40

In [40]:
```
l1 = l.copy()
l1
```

Out[40]: [10, 20, 30, 40, 40]

In [42]:
```
l1
```

Out[42]: [10, 20, 30, 40, 40]

In [44]:
```
l == l1
```

Out[44]: True

In [46]:
```
print(len(l))
print(len(l1))
```

```
5
5
```

In [48]:
```
l1
```

Out[48]: [10, 20, 30, 40, 40]

In [50]:
```
l1.append(2.3)
l1.append(True)
l1.append(1+2j)
```

In [52]:
```
l1
```

Out[52]: [10, 20, 30, 40, 40, 2.3, True, (1+2j)]

In [54]:
```
l
```

Out[54]: [10, 20, 30, 40, 40]

In [56]:
```
l1.count(20)
```

Out[56]: 1

In [58]:
```
l1.count(40)
```

Out[58]: 2

In [60]:
```
l.count(30)
```

Out[60]: 1

In [62]:
```
l1
```

Out[62]: [10, 20, 30, 40, 40, 2.3, True, (1+2j)]

In [64]:
```
l
```

Out[64]: [10, 20, 30, 40, 40]

In [66]:
```
l.count(40)
```

Out[66]: 2

In [72]:
```
l2=l1.copy()
```

In [74]:
```
l2
```

Out[74]: [10, 20, 30, 40, 40, 2.3, True, (1+2j)]

In [76]:
```
l2.remove(40)
```

In [78]:
```
l2
```

Out[78]: [10, 20, 30, 40, 2.3, True, (1+2j)]

In [80]:
```
l2.remove(True)
```

In [82]:
```
l2
```

Out[82]: [10, 20, 30, 40, 2.3, (1+2j)]

In [84]:
```
l2.remove(1+2j)
```

In [86]:
```
l2
```

Out[86]: [10, 20, 30, 40, 2.3]

In [88]:
```
l2.remove(2.3)
```

In [90]:
```
l2
```

Out[90]: [10, 20, 30, 40]

In [94]:
```
l2.clear()
```

In [96]:
```
l2
```

Out[96]: []

In [98]:
```
l
l1
```

Out[98]: [10, 20, 30, 40, 40, 2.3, True, (1+2j)]

In [100…
```
print(l)
print(l1)
```

```
[10, 20, 30, 40, 40]
[10, 20, 30, 40, 40, 2.3, True, (1+2j)]
```

In [104…
```
for i in l:
 print(i)
```

```
10
20
30
40
40
```

In [106…
```
l.append([1,2,3,'hi']) #nested list
l
```

Out[106… `[10, 20, 30, 40, 40, [1, 2, 3, 'hi']]`

In [134…
```
l.remove(40) # Remove the element directky not index wise
```

In [110…
```
l
```

Out[110… `[10, 20, 30, 40, [1, 2, 3, 'hi']]`

In [112…
```
l[:]
```

Out[112… `[10, 20, 30, 40, [1, 2, 3, 'hi']]`

In [114…
```
l[3]
```

Out[114… `40`

In [116…
```
l.pop()
l
```

Out[116… `[10, 20, 30, 40]`

In [118…
```
l1
```

Out[118… `[10, 20, 30, 40, 40, 2.3, True, (1+2j)]`

In [132…
```
l1.pop() # remove the last element from the list index wise
l1
```

Out[132… `[10, 20, 40]`

In [122…
```
l1.pop()
l
```

Out[122… `[10, 20, 30, 40]`

In [124…
```
l1
```

Out[124… `[10, 20, 30, 40, 40, 2.3]`

In [130…
```
l1.pop(2)
```
Out[130… 30

In [136…
```
l1
```
Out[136… [10, 20, 40]

In [138…
```
l
```
Out[138… [10, 20, 30]

In [140…
```
l1.insert(3,70)
l1
```
Out[140… [10, 20, 40, 70]

In [142…
```
l1.insert(1,15)
l1
```
Out[142… [10, 15, 20, 40, 70]

In [144…
```
l
```
Out[144… [10, 20, 30]

In [150…
```
l.insert(0,5) # it will insert value as per index passed in first argument
```
In [148…
```
l
```
Out[148… [5, 10, 20, 30]

In [152…
```
l1
```
Out[152… [10, 15, 20, 40, 70]

In [154…
```
l
```
Out[154… [5, 5, 10, 20, 30]

In [158…
```
l2.extend(l1)
```
In [160…
```
l1
```
Out[160… [10, 15, 20, 40, 70]

In [162…
```
l2
```
Out[162… [10, 15, 20, 40, 70]

In [164…
```
l1
```

Out[164… [10, 15, 20, 40, 70]

In [166…
```
l1
```

Out[166… [10, 15, 20, 40, 70]

In [168…
```
l2
```

Out[168… [10, 15, 20, 40, 70]

In [170…
```
l2.insert(3,90)
```

In [172…
```
l1
```

Out[172… [10, 15, 20, 40, 70]

In [174…
```
l2
```

Out[174… [10, 15, 20, 90, 40, 70]

In [182…
```
l2.extend(l1) # It will basically extend l2 list with l1 value
```

In [178…
```
l2
```

Out[178… [10, 15, 20, 90, 40, 70, 10, 15, 20, 40, 70]

In [180…
```
l1
```

Out[180… [10, 15, 20, 40, 70]

In [190…
```
l1.index(2) # Index function will show index position for that particular element p
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[190], line 1
----> 1 l1.index(2)

ValueError: 2 is not in list
```

In [192…
```
l1.index(15)
```

Out[192… 1

In [194…
```
l
```

Out[194… [5, 5, 10, 20, 30]

In [196…
```
l.index(5)
```

Out[196… 0

In [200…
```
l.index(20)
```

Out[200… 3

In [202…
```
l
```

Out[202… [5, 5, 10, 20, 30]

In [204…
```
l1
```

Out[204… [10, 15, 20, 40, 70]

In [206…
```
l1.insert(0,64)
```

In [208…
```
l1
```

Out[208… [64, 10, 15, 20, 40, 70]

In [210…
```
l1.sort() # sorting the list element in asc order
```

In [212…
```
l1
```

Out[212… [10, 15, 20, 40, 64, 70]

In [218…
```
l1.sort(reverse=True) # sorting the element in desc order
```

In [220…
```
l1
```

Out[220… [70, 64, 40, 20, 15, 10]

In [244…
```
l6 = [3, 5.6, 'a', 1+2j]
```

In [246…
```
l6.sort()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[246], line 1
----> 1 l6.sort()

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

In [248…
```
l5 = ['z', 'm', 'n', 'b']
l5
```

Out[248… ['z', 'm', 'n', 'b']

In [256…
```
l5.sort()
l5
```

Out[256… ['b', 'm', 'n', 'z']

In [258…
```
l1
```

Out[258… [70, 64, 40, 20, 15, 10]

In [260…
```
l1.reverse()
l1
```

Out[260…
```
[10, 15, 20, 40, 64, 70]
```

In [262…
```
l2
```

Out[262…
```
[10, 15, 20, 90, 40, 70, 10, 15, 20, 40, 70, 10, 15, 20, 40, 70]
```

In [264…
```
l2.reverse()
```

In [266…
```
l2
```

Out[266…
```
[70, 40, 20, 15, 10, 70, 40, 20, 15, 10, 70, 40, 90, 20, 15, 10]
```

In [268…
```
l2.remove(90)
l2
```

Out[268…
```
[70, 40, 20, 15, 10, 70, 40, 20, 15, 10, 70, 40, 20, 15, 10]
```

In [ ]: