list comprehension

efficient way of working with list,comprehension means ability to understand read the data from list instead of traditional way of reading

In [8]:

```
lst=[10,30,30,40,50,60,70,-2,-40,-6,-65,-98]
poslist=[i for i in lst if i>0]
neglist=[i for i in lst if i<0]
print('{} is post list'.format(poslist))
print('{} is neg list'.format(neglist))
```

```
[10, 30, 30, 40, 50, 60, 70] is post list
[-2, -40, -6, -65, -98] is neg list
```

In [14]:

```
# set comprehension
lst=[10,30,30,40,50,60,70,-2,-40,-6,-65,-98]
poslist=[i for i in lst if i>0]
neglist=[i for i in lst if i<0]
print('{} is post list'.format(poslist),type(poslist))
print('{} is neg list'.format(neglist),type(neglist))
print('=========   set section   ===========')
s=set(poslist)
s2=set(neglist)
print(s,type(s))
print(s,type(s))
```

```
[10, 30, 30, 40, 50, 60, 70] is post list <class 'list'>
[-2, -40, -6, -65, -98] is neg list <class 'list'>
=========   set section   ===========
{70, 40, 10, 50, 60, 30} <class 'set'>
{70, 40, 10, 50, 60, 30} <class 'set'>
```

In [67]:

```
#dict comprehension
tpl=(1,3,4,5,6,6,5,7,4)
d=dict([(val,val**2) for val in tpl])
for n,sn in d.items():
    print('{} \t\t{}'.format(n,sn))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[67], line 5
      3 d=dict([(val,val**2) for val in tpl])
      4 for n,sn in d.items():
----> 5     print('{} \t\t{}'.format(n,sn))
```

```
TypeError: 'str' object is not callable
```

```
In [ ]:
```