In [1]:
```
s1 = {1,2,3,4,20,30,100}
s2= {30,100,"a","b","c"}
```

In [4]:
```
s1.difference(s2)
s2.difference(s1)
```

Out[4]:
```
{'a', 'b', 'c'}
```

In [6]:
```
print(s1.symmetric_difference(s2))
s2.symmetric_difference(s1)
```

```
{1, 2, 3, 4, 'c', 20, 'b', 'a'}
```

Out[6]:
```
{1, 2, 20, 3, 4, 'a', 'b', 'c'}
```

In [7]:
```
print(s1)
print(s2)
```

```
{1, 2, 3, 4, 100, 20, 30}
{100, 'c', 'a', 'b', 30}
```

In [8]:
```
s1.symmetric_difference_update(s2)
```

In [9]:
```
print(s1)
```

```
{1, 2, 3, 4, 'c', 20, 'a', 'b'}
```

In [11]:
```
print(dir(s1))
```

```
['__and__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__gt__', '__hash__', '__
iand__', '__init__', '__init_subclass__', '__ior__', '__isub__', '__iter__', '_
_ixor__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__or__', '__rand
__', '__reduce__', '__reduce_ex__', '__repr__', '__ror__', '__rsub__', '__rxor_
_', '__setattr__', '__sizeof__', '__str__', '__sub__', '__subclasshook__', '__x
or__', 'add', 'clear', 'copy', 'difference', 'difference_update', 'discard', 'i
ntersection', 'intersection_update', 'isdisjoint', 'issubset', 'issuperset', 'p
op', 'remove', 'symmetric_difference', 'symmetric_difference_update', 'union',
'update']
```

```
### Tasks

1. l = [3,1,5,10,4,2,"h","s","b","d"]
output: [1,2,3,4,5,10,"b","d","h","s"]
2. take two inputs
name, phone number
d = {"KITS":"9883452147","APSSDC":"8434344342"}
contact updated successfully
contact added successfully
```

In [12]:
```
l = [3,1,5,10,4,3,10,2,"b","h","s","b","d"]
# [1,2,3,4]
```

In [14]:
```
l = [3,1,5,10,4,3,10,2,"b","h","s","b","d"]
l1 = []
for i in l: # 3
    if i not in l1:
        l1.append(i)
print(l1)

```

```
[3, 1, 5, 10, 4, 2, 'b', 'h', 's', 'd']
```

In [19]:
```
tuple(set(l))
```

Out[19]:
```
(1, 2, 3, 4, 5, 'd', 10, 's', 'h', 'b')
```

Comprehensions

- list comprehension
- set comprehension
- dictionary compresion

  Main advantage: reduce the number of lines of code

```
#### List comprehension

syntax:[output loop]  ===> there is no condtion<br>
[output loop condtion]   ===> only we have single condition<br>
[output if condition else loop] ==>if you have both if and else
```

In [22]:
```
l = [1,2,3,4,5,6,7]  #[2,4,6]
l1 = []
for i in l:
    if i%2==0:
        l1.append(i)
print(l1)
```

```
[2, 4, 6]
```

In [23]:
```
[i for i in l if i%2==0]
```

Out[23]:
```
[2, 4, 6]
```

In [24]:
```
["Even" if i%2==0 else "odd" for i in l]
```

Out[24]:
```
['odd', 'Even', 'odd', 'Even', 'odd', 'Even', 'odd']
```

In [ ]:
```
# Set comprehension
{"Even" if i%2==0 else "odd" for i in l}
```

Dictionary comprehension

syntax:{output(key:value) loop} ===> there is no condtion
{output(key:value) loop condtion} ===> only we have single condition
{output(key:value) if condition else loop} ==>if you have both if and else

In [28]:
```
l = [3,1,5,10,4,3,10,2,"b","h","s","b","d"]
#{3:2,1:1,5:1,10:1,2:1}
```

In [30]:
```
{i:l.count(i) for i in l if str(i).isdigit()}
```

Out[30]:
```
{3: 2, 1: 1, 5: 1, 10: 2, 4: 1, 2: 1}
```

In [ ]:
```

```

In [ ]:
```

```