Set-DataStructure

--add() --clear() --copy() --union() --update() --pop() --remove() --difference() --symmetric_diffrence() --intersection() --discard() --del --for loop --enumerate function in loop

In [ ]:
```
s={}     # define empty variable
s
```

In [ ]:
```
s=set()    # set type of variable as set
```

In [7]:
```
s
```

Out[7]:
```
set()
```

In [9]:
```
type(s)   # check type of variable
```

Out[9]:
```
set
```

In [11]:
```
s1={1,2,3,4,5,6,}
```

In [13]:
```
s1
```

Out[13]:
```
{1, 2, 3, 4, 5, 6}
```

In [15]:
```
s1.add(7)    # add value in set and this will add in order
```

In [17]:
```
s1
```

Out[17]:
```
{1, 2, 3, 4, 5, 6, 7}
```

In [19]:
```
s1.add(0)
```

In [21]:
```
s1
```

Out[21]:
```
{0, 1, 2, 3, 4, 5, 6, 7}
```

In [25]:
```
s=s1.copy()   # copy function not take any argument here thats why we assing whole
```

In [27]:
```
s
```

Out[27]:
```
{0, 1, 2, 3, 4, 5, 6, 7}
```

In [29]:
```
s1.clear()   # clear function clear all element from set
s1
```

Out[29]:
```
set()
```

In [34]:
```
s1=s.copy()
```

In [36]:
```
s1
```

Out[36]:
```
{0, 1, 2, 3, 4, 5, 6, 7}
```

In [38]:
```
s1.pop()   # pop function remove element from start index wise
```

Out[38]:
```
0
```

In [42]:
```
s1.pop(1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[42], line 1
----> 1 s1.pop(1)

TypeError: set.pop() takes no arguments (1 given)
```

In [46]:
```
s2={'nit',(1+2j),1,2,4,5}
```

In [48]:
```
s2
```

Out[48]:
```
{(1+2j), 1, 2, 4, 5, 'nit'}
```

In [50]:
```
s2.pop()
```

Out[50]:
```
'nit'
```

In [52]:
```
s3={1,2,3,4,5}     # declare three variable and assing value of each set
s4={4,5,6,7}
s5={7,8,9,10}
```

In [54]:
```
s3.difference(s4)     # here we check with diffrence function in s3 set and s4 set
```

Out[54]:
```
{1, 2, 3}
```

In [56]:
```
s4.difference(s3)
```

Out[56]:
```
{6, 7}
```

In [58]:
```
s3-s4             # difference function we denote as - sign as well
```

Out[58]:
```
{1, 2, 3}
```

In [60]:
```
s3.intersection(s4)   # here intersection function we used for common element show
```

Out[60]:
```
{4, 5}
```

In [62]:
```
s4.intersection(s3)
```

Out[62]:
```
{4, 5}
```

In [64]:
```
s3.symmetric_difference(s4)    # symmetric_difference function show not match eleme
```

Out[64]:
```
{1, 2, 3, 6, 7}
```

In [66]:
```
s3.discard(1)     # discard function removes given element from set
```

In [68]:
```
s3
```

Out[68]:
```
{2, 3, 4, 5}
```

In [70]:
```
s3.discard(6)     # in this example we dont have 6 value elemnt in set but discard
```

In [72]:
```
s3
```

Out[72]:
```
{2, 3, 4, 5}
```

In [74]:
```
s3.remove(5)    # remove function also remove element from set
```

In [76]:
```
s3
```

Out[76]:
```
{2, 3, 4}
```

In [78]:
```
s3.remove(6)    # here we dont have 6 value in set thats why remove function throws
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[78], line 1
----> 1 s3.remove(6)

KeyError: 6
```

In [80]:
```
s3.discard()     # discard function take one argument
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[80], line 1
----> 1 s3.discard()

TypeError: set.discard() takes exactly one argument (0 given)
```

In [82]:
```
s3.remove()   # remove function also take one argument
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[82], line 1
----> 1 s3.remove()

TypeError: set.remove() takes exactly one argument (0 given)
```

In [84]:
```
s3.union(s)     #  join two set but not take duplicate values
```

Out[84]:
```
{0, 1, 2, 3, 4, 5, 6, 7}
```

In [86]:
```
s
```

Out[86]: {0, 1, 2, 3, 4, 5, 6, 7}

In [88]:
```
s3
```

Out[88]: {2, 3, 4}

In [90]:
```
s3.update(s)   # update one set values into another set but no duplicate values all
```

In [92]:
```
s3
```

Out[92]: {0, 1, 2, 3, 4, 5, 6, 7}

In [94]:
```
s3.union(s4,s5)      # takes multiple parameter
```

Out[94]: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [96]:
```
s3.update(s4,s5)   # takes multiple parameter
```

In [98]:
```
s3
```

Out[98]: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [100…
```
for i in s3:
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
```

In [102…
```
for i in enumerate(s3):
    print(i)
```

```
(0, 0)
(1, 1)
(2, 2)
(3, 3)
(4, 4)
(5, 5)
(6, 6)
(7, 7)
(8, 8)
(9, 9)
(10, 10)
```

In [104…
```
s
```

Out[104…
```
{0, 1, 2, 3, 4, 5, 6, 7}
```

In [106…
```
del s
```

In [108…
```
s
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[108], line 1
----> 1 s

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

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: