Set Methods in python

Joining Sets

Sets in python more or less work in the same way as sets in mathematics. We can perform operations like union and intersection on the sets just like in mathematics.

1. Union () and Update ():

The union and update methods prints all items that are present in the two sets. The union () method returns a new set whereas update () method adds items into the existing set from another set.

Example:

```
s1={1,2,5,6}
s2={3,6,7}
s3=s1.union(s2)
print(s3)
```

Example2:

```
#print s1 before update
print(s1)
s1.update(s2)
print(s1)
#s1 set is update
```

2. Intersection and intersection update ():

The intersection () and intersection update () methods print only items that are similar to both the sets. The intersecting () method returns a new set whereas intersection update (). Method updates into the existing set from another set.

Example:

```
name1={"Munawar","ali","kamal",148}
name2={"Ali","Munawar","kamal",1}
intersection_name=name1.intersection(name2)
print(intersection_name)
```

Example2:

```
#before update
print(name2)
#after update
name2.update(name1)
print(name2)
```

Symmetric difference and symmetric update ():

The Symmetric difference () and Symmetric difference update () methods print only items that are different to both the sets. The Symmetric difference () method returns a new set whereas Symmetric different update (). Method updates into the existing set from another set.

Example:

```
#Symmetric difference and symmetric difference update
set1={12,11,23,11,121}
set2={21,11,12,122,1}
set3=set1.symmetric_difference(set2)
print(set3)
```

Example2:

```
print(set3)
#print before update
print(set1)
#print update symmetric difference update
set1.symmetric_difference_update(set2)
print(set1)
```

Difference and difference update ():

The difference () and difference update () methods prints only items that are only present in the original set and not in both the sets. The difference () method return a new set whereas difference update () method updates into the existing set from another set.

Example:

```
city1={"Karachi","Skardu","Kmg","Lahore"}
city2={"Skd","Karachi","Kmg","Multan"}
city3=city1.difference(city2)
print(city3)
```

Example2:

```
#city2 before update
print(city2)
city2.difference_update(city1)
#after difference update
print(city2)
```

Set Methods

There are several in both in built methods used for the manipulation of set. They are explained below.

Is Disjoint ():

The is disjoint method checks if items of the given set are present in another set. This method returns False if items are present, else it returns True.

Example:

```
item1={"bag","cake","bat"}
item2={"dog","cat","red"}
disjoint=item1.isdisjoint(item2)
```

```
print(disjoint)
```

Is Superset ():

The is disjoint method checks if all items of a particular set are present in the original set. It returns True if all the items are present, else it returns False.

Example:

```
setno1={"bag","cake","bat"}
setno2={"dog","cat","red","cake","bag","bat"}
issupperSet=setno1.issuperset(setno2)
print(issupperSet)
```

add ():

If you want to add a single item to the set use the add () method.

Example:

```
course={"Java","python","C++"}
#before adding javaScript
print(course)
course.add("JavaScript")
#affter adding javaScipt
print(course)
```

update ():

If you want to add more than one item, simply create another set or any other iterable object (list, tuple, dictionary), and use the update () method to add it into the existing set.

Example:

```
Set1={2,4,5,6,2,1}
Set2={3,5,6,7,8,10}
```

```
#before update
print(Set1)
Set1.update(Set2)
#after update
print(Set1)
```

remove () / Discord ():

We can use remove () and discord () methods to remove items form set.

Example:

```
Set5={2,4,5,6,2,1}
#before remove
print(Set5)
Set5.remove(5)
#after remove
print(Set5)
```

Discord raised the error if the items is not present in the set.

Example:

```
Set5={2,4,5,6,2,1}
#before remove
print(Set5)
Set5.remove(5)
#after remove
print(Set5)
Set5.remove(148)
print(Set5)
```

Output:

Set5.remove(148)

KeyError: 148

PS C:\Users\Latif Computers>

pop ():

This method removes the last item of the set but the catch is that we don't know which item gets popped as sets are undefined , However you can access the popped item if you assign the pop () method to a variable.

Example:

```
student={"ALI","Kamal","Zubair","Munawar"}
std_pop=student.pop()
print(student)
print(std_pop)
```

del ():

del is not a method, rather it is a keyword which deletes the set entirely.

Example:

```
del_student={"ALI","Kamal","Zubair","Munawar"}
del del_student
print(del_student)
```

clear ():

This method clear all items in the set and prints an empty set.

Example:

```
clear_name={"Ahmed","Munawar","Ameer","Qasim"}
clear_name.clear()
print(clear_name)
```

Check if item exists:

You can also check if an item exists in the set or not.

Example:

```
check_list={"Munawar","Raziq","Kamal","Ali","MaRaj"}
```

```
if "Munawar" in check_list:
    print("Munawar Is present in list")
else:
    print("Munawar is not present in list")
```

Source Code

```
# s1={1,2,5,6}
# s2={3,6,7}
# s3=s1.union(s2)
# print(s3)
# #print both set s1 and s2
# print(s1,s2)

# #print s1 before update
# print(s1)
# s1.update(s2)
# print(s1)
# #s1 set is update

# name1={"Munawar","ali","kamal",148}
# name2={"Ali","Munawar","kamal",1}
# intersection_name=name1.intersection(name2)
# print(intersection_name)

# #before update
# print(name2)
# #after update
# name2.update(name1)
# print(name2)


#Symmetric difference an symmetric difference update
# set1={12,11,23,11,121}
# set2={21,11,12,122,1}
# set3=set1.symmetric_difference(set2)
# print(set3)
# #print before update
# print(set1)
# #print update symmetric difference update
# set1.symmetric_difference_update(set2)
# print(set1)
```

```
# city1={"Karachi","Skardu","Kmg","Lahore"}
# city2={"Skd","Karachi","Kmg","Multan"}
# city3=city1.difference(city2)
# print(city3)
# #city2 before update
# print(city2)
# city2.difference_update(city1)
# #after difference update
# print(city2)

# item1={"bag","cake","bat"}
# item2={"dog","cat","red"}
# disjoint=item1.isdisjoint(item2)
# print(disjoint)

# setno1={"bag","cake","bat"}
# setno2={"dog","cat","red","cake","bag","bat"}
# issupperSet=setno1.issuperset(setno2)
# print(issupperSet)

# course={"Java","python","C++"}
# #before adding javaScript
# print(course)
# course.add("JavaScript")
# #affter adding javaScipt
# print(course)


# Set1={2,4,5,6,2,1}
# Set2={3,5,6,7,8,10}
# #before update
# print(Set1)
# Set1.update(Set2)
# #after update
# print(Set1)



# Set5={2,4,5,6,2,1}
# #before remove
# print(Set5)
# Set5.remove(5)
# #after remove
# print(Set5)
# Set5.remove(148)
```

```
# print(Set5)

# student={"ALI","Kamal","Zubair","Munawar"}
# std_pop=student.pop()
# print(student)
# print(std_pop)

# del_student={"ALI","Kamal","Zubair","Munawar"}
# del del_student
# print(del_student)

# clear_name={"Ahmed","Munawar","Ameer","Qasim"}
# clear_name.clear()
# print(clear_name)

check_list={"Munawar","Raziq","Kamal","Ali","MaRaj"}
if "Munawar" in check_list:
    print("Munawar Is present in list")
else:
    print("Munawar is not present in list")
```

Thank You