Reviving INDIA

Transforming Lives Digitally

Python sets

-Python Series

Introduction

- A Set is an unordered collection data type that is iterable, mutable and has no duplicate elements.
- A set is a collection which is both *unordered* and *unindexed*.
- Sets are written with curly brackets.

Create

- The set can be created by enclosing the comma-separated items with the curly braces {}.
- Python also provides the set() method, which can be used to create the set by the passed sequence.
- But mutable elements (list, dictionary, set) can't be a member of set.

Cont...

```
myset = {"revin", "reviving", 54, 54.36}
print(myset)

x = set([1,2,3,4])
print(x,type(x))
```

Empty Set

- Creating an empty set is a bit different because empty curly {} braces are also used to create a dictionary as well.

- So Python provides the set() method used without an argument to create an empty set.

Cont..

```
# Empty set using set() function
myset = set()
print(type(myset))
```

Access

- In set we can't access using index or key only we can loop to get the elements or items of set.

```
myset = {"revin", "reviving", 54, 54.36}
for var in myset:
        print(myset)
```

Add or Change

- We cannot change its items, but you can add new items.
- We can add new items with add() & update() method

Add() method

- to add one item to a set use the add() method.

myset.add(“hello world")

print(myset)

Update method()

- To add items from another set into the current set, use the update() method.

myset1 = {"revin", "reviving", 54, 54.36}
myset2 = {1, 2}

myset1.update(myset2)

Note : update method can take any iterable object

Remove

- To remove an item in a set, use following methods:

- remove
- discared
- pop
- clear
- del

Join Sets

- union()-pipe (|) operator

This method that returns a new set containing all items from both sets.

```
myset1 = {"revin", "reviving", 54, 54.36}

myset2 = {1,2,3}

set3 = myset1.union(myset2)
print(set3)
```

- The update() method inserts the items in set2 into set1:

```
myset1 = {"revin", "reviving", 54, 54.36,1}

myset2 = {1,2,3}

set3 = myset1.update(myset2)
print(myset1)
```

Note :Both union() and update() will exclude any duplicate items.

- intersection_update() : method will keep only the items that are present in both sets.
- x.intersection_update(y)
- it modifies the original set

- intersection() **&** operator: method will return a *new* set, that only contains the items that are present in both sets.

z = x.intersection(y)

intersection() method returns a new set.

- symmetric_difference_update(): method will keep only the elements that are NOT present in both sets.

- x.symmetric_difference_update(y)

Difference between the two sets

- by using the subtraction (-) operator or difference method.

```
Day1 = {"Mon",  "Tues", "Wed", "Thurs"}
Day2 = {"Mon", "Tues", "Sun"}
print(Day2.difference(Day1))
```