DICTIONARY IN PYTHON

Dictionaries are used to store data values in key:value pairs

“key” : value

They are unordered, mutable(changeable) & don’t allow duplicate key

Syntax:

Attribute_name={“keys”:value,”Keys”:value}

Ex: Person={“Name”:”Divya”,”salary”:25000}

Ex1:WAP using Dictionary store data Keys and value form

```
student={
    "id":34,
    "name":"Riya",
    "class":6,
    "marks":[67,78,69,56]
}
print(student)  #To print all Keys and Value
print("_____")
    #To print value of specific key use syntax= print("Keyname")
print("ID  =",student["id"])
print("Name =",student["name"])
print(student["class"])+
print(student["marks"])
\ #to assign or add new data
student["Address"]="Nashik"
print(student)  #to print updated new dictionary
```

```
{'id': 34, 'name': 'Riya', 'class': 6, 'marks': [67, 78, 69, 56]}
_____
ID  = 34
Name = Riya
6
[67, 78, 69, 56]
{'id': 34, 'name': 'Riya', 'class': 6, 'marks': [67, 78, 69, 56],
'Address': 'Nashik'}
```

DICTIONARY METHODS

```
myDict.keys( )                  #returns all keys

myDicts.values()                #return all value

myDict.items( )                 #returns all pairs as tuples

myDict.update( newDict )        #inserts the specified items to the dictionary

myDict.get( “key““ )            #return  keys  the   according to value
```

USING DICTIONARY METHODS

```
student={
    "id":34,
    "name":"Riya",
    "class":6,
    "marks":[67,78,69,56]
}
print("-----------------Dictionary Methods-----------------")
print(student.keys())
print(student.values())
print(student.items())
print(student.get("Keys"))
print(student.update({"mobile_no":"938498349"}))
print(student)
```

```
dict_keys(['id', 'name', 'class', 'marks', 'Address'])
dict_values([34, 'Riya', 6, [67, 78, 69, 56], 'Nashik'])
dict_items([('id', 34), ('name', 'Riya'), ('class', 6), ('marks', [67, 78, 69, 56]), ('Address', 'Nashik')])
None
None
{'id': 34, 'name': 'Riya', 'class': 6, 'marks': [67, 78, 69, 56], 'Address': 'Nashik', 'mobile_no': '938498349'}
```

SET IN PYTHON

Set is the collection of the unordered items.

Each element in the set must be unique & immutable.

null_set = set( ) #empty set

syntax nums = { 1, 2, 3, 4 }

set2 = { 1, 2, 2, 2 } #repeated elements stored only once, so it resolved to {1, 2}

null_set = set( ) #empty set syntax

SET METHODS

set.add(el)	#adds an element
set.remove(el)	#removes the element
set.clear( )	#empties the set
set.pop( )	#removes a random number
set.union(set2)	#combines both set values & returns new
set.intersection(set2)	# Combining comman value an return new value

Add Any Iterable

The object in the `update()` method does not have to be a set, it can be any iterable object (tuples, lists, dictionaries etc.)

SETS METHOD

set.union(set2) #combines both set values & returns new

set.intersection(set2) # Combining comman value an return new value

```
a={1,2,3,4,4,5,6,30,32}
b={30,50,60,4,5,6,1}
print(a.union(b)) #printvalue of a and b
print(a.intersection(b))   #only print comman values in both set a and b
```

```
{32, 1, 2, 3, 4, 5, 6, 50, 60, 30}
{1, 4, 5, 6, 30}
```

WAP to join multiple SET using “ | ”(JOIN)

```
set1 = {"a", "b", "c"}
set2 = {1, 2, 3}
set3 = {"John", "Elena"}
set4 = {"apple", "bananas", "cherry"}

myset = set1 | set2 | set3 |set4
print(myset)
```

o/p
{banana, 2, apple, 'b', 3, 'a', 1, cherry, 'c', Elena, John}

Join a Set and a Tuple

The `union()` method allows you to join a set with other data types, like lists or tuples.

```
x = {"a", "b", "c"}
y = (1, 2, 3)
z = x.union(y)
print(z)
```

O/P {'b', 1, 'c', 2, 3, 'a'}

The & operator only allows you to join sets with sets, and not with other data types like you can with the intersecton() method.

INTERSECTION IN SET

```
set1 = {"apple", "banana", "cherry"}
set2 = {"google", "microsoft", "apple"}
set3 = set1.intersection(set2)
print(set3)
```