**9.5. SET**

Mathematically a **set** is a collection of items **not in any particular order**. A Python set is similar to this mathematical definition with below additional conditions.

* The elements in the set **cannot be duplicates**.(**must be unique**)
* The elements in the set are **immutable**(cannot be modified) but the set as a whole is mutable. Overall set is **mutable**
* Maintains **no order.**
* There is **no index** attached to any element in a python set. So they do not support any indexing or slicing operation.

## Set Operations

The sets in python are typically used for mathematical operations like union, intersection, difference and complement etc. We can create a set, access it’s elements and carry out these mathematical operations as shown below.

## Creating:

A set is created by using the set() function or placing all the elements within a pair of curly braces.

days= set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])

months= {"Jan", "Feb", "Mar"}

dates= {21, 22, 17}

print(days)

print(months)

print(dates)

1. **Retrieving:**

We cannot access individual values in a set. We can only access all the elements together as shown above. But we can also get a list of individual elements by looping through the set.

days=set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])

for d in days:

print(d)

**3.Updating:**

We can add elements to a set by using add() method. Again as discussed there is no specific index attached to the newly added element.

days=set(["Mon","Tue","Wed","Thu","Fri","Sat"])

days.add("Sun")

print(days)

Output :{'Wed','Sun','Fri','Tue','Mon','Thu','Sat'}

**4.Deleting:**

We can remove elements from a set by using discard() method. Again as discussed there is no specific index attached to the newly added element.

days=set(["Mon","Tue","Wed","Thu","Fri","Sat"])

days.discard("Sun")

print(days)

## Union:

## The union operation on two sets produces a new set containing all the distinct elements from both the sets. In the below example the element “Wed” is present in both the sets.

DaysA=set(["Mon","Tue","Wed"])

DaysB=set(["Wed","Thu","Fri","Sat","Sun"])

AllDays=DaysA | DaysB

print(AllDays)

## Intersection

The intersection operation on two sets produces a new set containing only the common elements from both the sets. In the below example the element “Wed” is present in both the sets.

DaysA=set(["Mon","Tue","Wed"])

DaysB=set(["Wed","Thu","Fri","Sat","Sun"])

AllDays=DaysA & DaysB

print(AllDays)

## Difference

## The difference operation on two sets produces a new set containing only the elements from the first set and none from the second set. In the below example the element “Wed” is present in both the sets so it will not be found in the result set.

DaysA=set(["Mon","Tue","Wed"])

DaysB=set(["Wed","Thu","Fri","Sat","Sun"])

AllDays=DaysA - DaysB

print(AllDays)

## Compare

## We can check if a given set is a subset or superset of another set. The result is True or False depending on the elements present in the sets.

DaysA=set(["Mon","Tue","Wed"])

DaysB=set(["Mon","Tue","Wed","Thu","Fri","Sat","Sun"])

SubsetRes=DaysA <= DaysB

SupersetRes=DaysB >= DaysA

print(SubsetRes)

print(SupersetRes)