**2.21 Python Set**

A set is an unordered collection of distinct objects. Functions related to sets can be used to find differences, intersections, and unions of lists, to check the presence of an element in a list, duplicate elements in a list etc. But sets do not support indexing, slicing, or other sequence-like behaviour. There are two built-in set types, [***set***](http://docs.python.org/library/stdtypes.html#set) and ***[frozenset](http://docs.python.org/library/stdtypes.html" \l "frozenset" \o "frozenset).*** The [set](http://docs.python.org/library/stdtypes.html#set) type is ***mutable***. The **[frozenset](http://docs.python.org/library/stdtypes.html" \l "frozenset" \o "frozenset)** type is immutable. That means, its contents cannot be altered after it is created; it can therefore be used as a dictionary key or as an element of another set. A set can be created as follows.

*set([1, 2, 7.6, 'Thomas'])*

A list can be converted into a set as follows.

*x=[1,2,3,4] to create a list*

*y=set(x) to create a set*

**Table of functions connected with set**

|  |  |
| --- | --- |
| *len(s)* | To find the number of elements in the set named s. |
| *x in s* | Test wheatear *x* is a member of s. Returns true if x is a member of s. |
| *x not in s* | Test wheatear *x* is not a member of s. Returns true, if x is not a member of s. |
| *s.add(x)* | add element x to set s |
| *s.remove(x)* | remove x from set s; raises an error   if not present |
| *s.union(t)* | new set with elements from both *s* and *t* |
| *s.intersection(t)* | new set with elements common to *s* and *t* |
| *s.difference(t)* | new set with elements in *s* but not in *t* |
| *s.symmetric\_*  *difference(t)* | new set with elements in either *s* or *t* but not both |