**Built-in data Structures in Python**

We have 4 built-in data types in Python

1.list

2.tuple

3.set

4.dictionary

**1.List:**

List is an ordered collection of items. List is enclosed in square brackets. lists are mutable each item is separated by comma.

List = [‘gopi’, 2.3, 4, ‘puropale’]

* For accessing list times we use indexing
* We can perform iterations on list
* We can perform Concatenation on lists
* Obtaining a part of list is called list slicing

Some of the list methods are:

1. Append
2. Extend
3. Insert
4. Remove
5. Pop
6. Clear
7. Sorting
8. Reverse
9. Copy
10. Index
11. len

**1.Append:**

Append adds item to the end of the list

list\_a = [1,'gopi',2.2,3]  
list\_a.append(20)  
print(list\_a)

**2.Extend:**

Adds all items to the end of the list

list\_b = [1, 'gopi', 2.4, 5]  
list\_b.extend([20,40])  
print(list\_b)

**3.Insert:**

Insert item at the specific location

list\_c = [1, 'gopi', 3.4, 20]  
list\_c.insert(1,'mani')  
print(list\_c)

**4.Remove:**

It removes the first occurrences of the item it raises error if the value is not found.

list\_d = [20, 'santosh', 50, 2.3,20]  
list\_d.remove(20)  
print(list\_d)

**5.Pop:**

Pop will remove the last items from the list .

list\_d = [20, 'gopi', 70, 100]  
list\_d.pop()  
print(list\_d)

**6.Clear:**

Clear will remove the all items from the list.

list\_e = [20, 'gopi', 70, 100]  
list\_e.clear()  
print(list\_e)

**7.Index:**

Returns the index of the first occurrences of the value if not found it will return an error.

list\_f = [11,'hello', 5.5, 20]  
print(list\_f.index('hello'))

**8.Count:**

Count returns the number of occurrences of a specified item

list\_g = [1,2,3,2,4,2,2,2,2,]  
print(list\_g.count(2))

**9. reverse :**

Prints the values in reverse order

list\_h = [1,2,3,4]  
list\_h.reverse()  
print(list\_h)

**10.length:**

Prints the length of the list

List\_i = [3,1,2,6,7]

Print(len(list\_i))

**Tuples:**

Tuple holds an ordered sequence of items. Tuple is immutable. items are enclosed in round brackets

1. count(item)
2. len(tuple)
3. max(tuple)
4. min(tuple)
5. sum(tuple)
6. sorted(tuple)
7. tuple(iterable)

**1.count(item)**

Returns the number of occurrences of the specified item.

tuple\_a = (1,2,3,4,2,2)  
print(tuple\_a.count(2))

**2. len(tuple)**

returns the length of the tuple.

tuple\_a = (1,2,3,4,2,2)

print(len(tuple\_a)

**3.max(tuple)**

Returns the max value from the tuple

tuple\_a = (1,2,3,4,2,2)

print(max(tuple\_a)

**4.min(tuple)**

Returns the min value from the tuple

tuple\_a = (1,2,3,4,2,2)

Print(min(tuple\_a))

**5.sum(tuple)**

Returns the sum of the tuple

tuple\_a = (1,2,3,4)

print(sum(tuple\_a))

**6.sorterd(tuple):**

It sorts the elements in the asc order and returns as the list

tup = (3, 1, 2)  
print(sorted(tup)) # [1, 2, 3]

**Set:**

Set is an unordered collection of items. Set is enclosed with curly braces. Set does not allows duplicate values set is mutable.

Some of the set operations are:

1. union
2. difference
3. intersection
4. symmetric difference

set comparisons:

* issubset ()
* issuperset ()
* isdisjoint ()

methods in set:

1. add(element)
2. update(iterable)
3. remove(element)
4. discard(element)
5. pop()
6. clear()

**union:**

union of set is a set containing all elements of both sets when we perform union operation on set it will sort the elements in a ASC order

**code:**

set\_a = {4, 2, 8}

set\_b = {1, 2}

union = set\_a | set\_b

print(union)

output:

{1,2,4,8}

**Intersection:**

Intersection set is a set containing common elements from both the set

**Code:**

set\_a = {4, 2, 8}

set\_b = {1, 2}

intersection = set\_a.intersection(set\_b)

print(intersection)

**output:**

{2}

**Difference:**

Difference set is a set contains all values from the first set.

**Code:**

set\_a = {4, 2, 8}

set\_b = {1, 2}

diff = set\_a - set\_b

print(diff)

**output:**

{8,4}

**Symmetric difference:**

Symmetric difference is a set containing elements which are not common to both the sets.

**Code:**

set\_a = {4, 2, 8}

set\_b = {1, 2}

symmetric\_diff = set\_a ^ set\_b

print(symmetric\_diff)

**output:**

{8,1,4}

**Set compressions:**

Set comparisons are used to validate whether one set fully exists within another

**issubset ()**

set\_1 = {'a', 1, 3, 5}

set\_2 = {'a', 1}

is\_subset = set\_2.issubset(set\_1)

print(is\_subset)

output:

true

**issuperset ()**

Returns True if all elements of second set are in first set. Else, False

set\_1 = {'a', 1, 3, 5}

set\_2 = {'a', 1}

is\_superset = set\_1.issuperset(set\_2)

print(is\_superset)

output:

true

**Disjoint Sets**

Returns True when they have no common elements. Else, False

set\_a = {1, 2}

set\_b = {3, 4}

is\_disjoint = set\_a.isdisjoint(set\_b)

print(is\_disjoint)

output:

true

1. **add(element)**

add single elements to the set if that element is already exists in the set it returns nothing

**code:**

s = {1,2,3,4}  
s.add(5)  
print(s)

output:

{1,2,3,4,5}

**2.update(iterable)**

Adds multiple elements to the sets. Ignores duplicate values.

**Code:**

a = {1,2}  
a.update([3,4,5])  
print(a)

**output:**

{1,2,3,4,5}

**3. remove(element)**

Removes the elements from the set. Raises an error if keyword is not found

**Code:**

n = {1,2,3,4}  
n.remove(3)  
print(n)

output:

{1, 2, 4}

**4.discard(element)**

Removes the elements from the set. Does not Raises an error if keyword is not found

**Code:**

n = {1,2,3,4}  
n.remove(5)  
print(n)

**output:**

{1, 2,3, 4}

**5.clear()**

Removes all the elements from the set.

**Code:**

b = {1,2,3,4}

b.clear()

print(b)

output:

set()

**Dictionary :**

Unordered collection of items is called dictionary. In dictionary items are in the form of key value pairs. Created by enclosing in curly braces. Key are immutable and must be unique. Values can be repeated.

**1.Adding and updating:**

Adds the new key value pair or updates the value

**Adding:**

**Code:**

dict = {'name': 'gopi','age': 23}  
dict['email'] = 'gopi99@gmail.com'  
print(dict)

**Output:**

{'name': ‘gopi', 'age': 23, 'email': 'gopi99@gmail.com'}

**Updateing:**

dict = {'name': 'gopi','age': 23}  
dict['name'] = ‘mani’  
print(dict)

**Output:**

{'name': ‘mani', 'age': 23, 'email': 'gopi99@gmail.com'}

**Deleting:**

Deleting an specific item from the list

Dict = {'name': ‘mani', 'age': 23, 'email': 'gopi99@gmail.com'}

Del Dict[‘name’]

Print(Dict)

**Output:**

{'age': 23, 'email': 'gopi99@gmail.com'}