Collection Types

By
Dr. B. Sangeetha
Assistant Professor (Sr. Gr.)
Department of Information Technology
PSG College of Technology

Collection Types

- Also called Compound Data types

- Four Types:
  - List
  - Tuple
  - Set
  - Dictionary

List

- Multiple items in a single variable
- Different data types
- Maintains insertion order
- Mutable – values in the list can be changed
- Allow duplicate values
- Written in square brackets []
- Can be indexed

Operations on List

- Create List

  var = [] **(or)** var = list()

- Access List
  - With index values – starts from 0
  - Negative indexing - from last item

- Remove
  - Element - del var[index]
  - Entire List - del var
- Length
  - len(var) ➔ returns number of elements
- Concatenate
  - list1 + list2

Contd/-

- Change / update
  - var[index] = value
  - Range of values to change - var[1:3] = [43, “san”]
  - **Check:** if more values are specified than the range
- Slicing
  - var[startindex : stop]
    - Ex: mylist[1:4]
      mylist [1: -3] ➔starts from index 1 and finds the length of the list -3
- Search
  - Print(“data” in var) ➔returns True or False based on existence
- Repetition
  - Var *2 --- repeat itself 2 times

Contd/-

- Packing - Assigning values to list
  - Var = [1, “hello”]
- UnPacking – Values of list assigned to variables
  - a,b = var ➔ Ensure number of values in list = to number of varaibles

List - Methods

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Tuple

- Multiple items in a single variable
- Different data types
- Maintains insertion order
- Immutable – values in the tuple cannot be changed [Doesn’t allow - add/remove/change]
- Allow duplicate values
- Written in brackets ()
- Can be indexed

Operations on Tuple

- Create Tuple

  var = () **(or)** var = tuple()

- Access Tuple
  - With index values – starts from 0
  - Negative indexing - from last item
- Length
  - len(var) ➔ returns number of elements
- Repetition
  - Var *2 --- repeat itself 2 times
- Concatenate
  - list1 + list2

Contd/-

- Slicing
  - Mylist[startindex : stop]
    - Ex: var[1:4]

      var [1: -3] ➔starts from index 1 and finds the length of the list -3
- Search
  - Print("data" in var) ➔returns True or False based on existence

- Packing - Assigning values to tuples
  - Var = (1, "hello")
- UnPacking – Values of tuples assigned to variables
  - a,b = var ➔ Ensure number of values in tuples = to number of varaibles

Tuple - Methods

Method	Description
count()	Returns the number of elements with the specified value
index()	Returns the index of the first element with the specified value

Dictionary

- Information as key-value pairs
- Keys – indexes (basic type and distinct)
- Value – can be any type (basic/compound)
- Mutable
- Ordered
- Keys - No duplicates
- Represented using { } – separate each key/value pair by ,

Operations on Dictionary

- Representation:

  thisdict = { “Rollno”: 20, “name” : “san”}

- Empty Dict:
  - Thisdict = dict()

- Access

  thisdict[“key”] → get value

- Add

  Thisdict[“key”] = value

  Already existing – will update /otherwise will add new entry

Contd/-

- Change
  - Thisdict[“key”] = value
- Remove
  - del thisdict[“key”]
  - del thisdict
- Copy
  - Dict1 = dict2 // shallow copy

Dictionary - Methods

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Set

- multiple items in a single variable
- unordered, and unindexed
- do not allow duplicate values
- Represented using { }

Operations on Dictionary

- Representation:

  thisset = {1,2,3,4}

  thisset = {1,2,3,4,1} # Repetitive elements

- Empty Set:
  - Thisdict =set()

Methods

- Add

set1 = {1,2,3}

set1.add(4) - > set1 = {1,2,3,4} # one element of basic type

- Add all elements in a set (or) list to existing set

set1 = {1,2,3}

set2 = {4,5,6}

set1.add(set2) - > set1 = {1,2,3,4,5,6}

- Access elements in a set

Only use iterable like for loop to access elements

Methods

- Remove

set1 = {1,2,3}

set1.remove(4) # error as element not in set

set1.discard(4) # doesn’t throw error

set1.pop() delete last element , order is not known

set1.clear() empties the set

Mathematical set operations

- Set operations

s1= {1,2,3} s2 = {1,4,5,6}

s3 = s1.union(s2) s3 -> {1,2,3,4,5,6}

s3 = s1.difference(s2) s3 -> {2,3}

s3 = s1.symmetric_difference(s2) s3->{2,3,4,5,6}

s3 = s1.intersection(s2) s3->{1}