1: The Python pop() method removes an item at a specified index value from a list. This method returns the item you have removed so you know what modification has been made to your list. pop() accepts one argument: the index of the item you want to remove.

The pop() function takes in an index value, then checks whether the item exists in the list. If the item does exist, the method will remove the item from the list and return the item it has removed. If the item does not exist, an error will be returned.

Let’s use an example to showcase the pop () function and how it removes and returns the item at the index passed through the function.

teas = ['Earl Gray', 'Oolong', 'English Breakfast', 'Chai', 'Chamomile']

remove\_earl\_gray = teas.pop (0)

print(remove\_earl\_gray)

print(teas)

**2: TEN LIST METHOD AND THEIR USES.**

1: insert (): Insert a specified object before the index.

my\_list=["Mango","Orange","Tomatoes","Banana","Water Melon","Lemon"]

my\_list. Insert(2,"Beans")

print(my\_list)

['Mango', 'Orange', 'Beans', 'Tomatoes', 'Banana', 'Water Melon', 'Lemon', 'Yam']

2. append (): The append () method is used to insert a new item at the end of the list.

my\_list=["Mango","Orange","Tomatoes","Banana","Water Melon","Lemon"]

my\_list. Append("Yam")

print(my\_list)

output: ['Mango', 'Orange', 'Tomatoes', 'Banana', 'Water Melon', 'Lemon', 'Yam']

3. remove (): The remove () method is used to remove a particular item from the list.

my\_list=["Mango","Orange","Tomatoes","Banana","Water Melon","Lemon"]

my\_list. remove(“Banana”)

print(my\_list)

['Mango', 'Orange', 'Tomatoes', 'Water Melon', 'Lemon']

4. extend():The extend() method is used to merge two list items and store the merged items in the first list.

my\_list=["Mango","Orange","Tomatoes","Banana","Water Melon","Lemon"]

my\_list.remove("Banana")

ingred=["salt","pepper","Magie","onion","garden Eggs"]

my\_list. extend(ingred)

print(my\_list)

['Mango', 'Orange', 'Tomatoes', 'Water Melon', 'Lemon', 'salt', 'peper', 'Magie', 'onion', 'garden Eggs']

5. count(): The count() method is used to count the number times that any given item appears in a list.

my\_list= ['Mango', 'Orange', 'Tomatoes', 'Mango', 'Water Melon', 'Lemon', 'salt', 'peper', 'Magie', 'onion', 'garden Eggs']

count\_list=my\_list.count("Mango")

print(count\_list)

2

6. index():The index() method is used to obtain the position value of any item in the list.

my\_list=["Mango","Orange","Tomatoes","Mango","Banana","Water Melon","Lemon"]

print (my\_list. Index("Lemon"))

output is 6

7. copy (): The copy () method is used to make a copy of a list. This method is useful for keeping original list values before modifying the list.

my\_list=["Mango","Orange","Tomatoes","Mango","Banana","Water Melon","Lemon"]

copy\_my\_list=my\_list. Copy ()

print(copy\_my\_list)

['Mango', 'Orange', 'Tomatoes', 'Mango', 'Banana', 'Water Melon', 'Lemon']

8. sort():The sort() method is used to sort list data. This method is useful when you are working with the same type of list data and you need to organize the data for any programming purpose.

my\_list=["Mango","Orange","Tomatoes","Mango","Banana","Water Melon","Lemon"]

print(my\_list)

my\_list. sort ()

print(my\_list)

['Mango', 'Orange', 'Tomatoes', 'Mango', 'Banana', 'Water Melon', 'Lemon']

['Banana', 'Lemon', 'Mango', 'Mango', 'Orange', 'Tomatoes', 'Water Melon']

9. reverse():The reverse() method is used to reverse the items in any list. This method is useful for sorting lists in descending order.

my\_list=["Mango","Orange","Tomatoes","Mango","Banana","Water Melon","Lemon"]

print(my\_list)

my\_list.reverse()

print(my\_list)

['Mango', 'Orange', 'Tomatoes', 'Mango', 'Banana', 'Water Melon', 'Lemon']

['Lemon', 'Water Melon', 'Banana', 'Mango', 'Tomatoes', 'Orange', 'Mango']

10. clear (): The clear () method is used to remove all the items in a list and to empty lists. This method is useful for re-assigning the values of a list by removing the previous items.

my\_list=["Mango","Orange","Tomatoes","Mango","Banana","Water Melon","Lemon"]

print(my\_list)

my\_list. Clear ()

print(my\_list)

['Mango', 'Orange', 'Tomatoes', 'Mango', 'Banana', 'Water Melon', 'Lemon']

[]

EIGHT TRDICTIONARY METHODS AND HOW TO USED THEM

A python dictionary is a collection data type which is wrapped in braces, {}, with a series of key value pairs inside the braces. Each key is connected to a value. We use a key to access the value associated with that key. A key can be a number, a string, a list, or even another dictionary.

1: keys():The method keys() returns a list of all the available keys in the dictionary.

*information= {"Name”: “Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}*

*print (information. Keys ())*

*output*

dict\_keys (['Name', 'contact', 'class', 'Duration', 'Teacher'])

2: items (): This method returns a list of dictionary's (key, value) as tuple.

*information= {"Name”: “Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}*

*print (information. items ())*

*output:*

dict\_items ([('Name', 'Zato Stephen'), ('contact', '0242047606'), ('class', 'Python'), ('Duration', 'Three months'), ('Teacher', 'Brooks')])

3: values (): This method returns list of dictionary dictionary's values from the key value pairs.

information= {"Name”: “Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

print (dict. values ())

dict\_values (['Zato Stephen', '0242047606', 'Python', 'Three months', 'Brooks'])

4: pop (): The method pop(key) Removes and returns the value of specified key.

information={"Name”: Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

print(information. Pop(“Name”)

output : Zato Stephen

5: copy (): This method Returns a shallow copy of dictionary.

information={"Name”: Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

new\_information=information. Copy ()

print(new\_information)

6: clear():The method clear() Removes all elements of dictionary.

information= {"Name”: Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

information. Clear()

print(information)

output: {}

7: get (): This method returns value of given key or None as default if key is not in the dictionary.

information= {"Name”: Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

print ('Name: ', information.get('Name'))

output: None

8: update (): The update () inserts new item to the dictionary.

information= {"Name”: Zato Stephen","contact":"0242047606","class":"Python","Duration":"Three months","Teacher":"Brooks"}

information. Update({'Python':3.9})

print(information)

output

{'contact': '0242047606', 'class': 'Python', 'Duration': 'Three months', 'Teacher': 'Brooks', 'Python': 3.9}

TEN TUPLE METHOD AND THEIR USES.

Tuples are used to store multiple items in a single variable. Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage. A tuple is a collection which is ordered and unchangeable. Tuples are written with round brackets.

Tuple Methods

Python has two built-in methods that you can use on tuples.

count() Returns the number of times a specified value occurs in a tuple

thistuple = (1, 3, 7, 8, 7, 5, 4, 6, 8, 5)

x = thistuple.count(5)

print(x)

2

index() Searches the tuple for a specified value and returns the position of where it was found

thistuple = (1, 3, 7, 8, 7, 5, 4, 6, 8, 5)

x = thistuple. index(8)

print(x)

3