**Dictionaries**

* Dictionaries are used to store data values in key: value pairs and can be referred to by using the key name.
* The values in dictionary items can be of any data type
* From Python's perspective, dictionaries are defined as objects with the data type 'dict'
* A dictionary is a collection of items which are ordered\*, changeable and do not allow duplicates with same key but values can be duplicated.
* If duplicates two pairs, overwrite first value with new value
* As of Python version 3.7, dictionaries are ordered. In Python 3.6 and earlier, dictionaries are unordered.
* Dictionaries are written with curly brackets, and have keys and values

**Create a dictionary**

print("Create and print a dictionary")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print(dictOne) *#{1: 'Horse', 2: 'Dog', 3: 'Cat', 4: 'Lion'}*

**Access the value** of a specified key by specifying the key or using get() method

print("Get the value of the \"4\" key")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print(dictOne[4]) *#'Lion'*

**get()-** returns a value of a specified key

print("Get the value of the \"4\" key using get()")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print(dictOne.get(4)) *#'Lion'*

**keys()-** returns all the keys in the dictionary

print("return all the keys")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print(dictOne.keys()) *#dict\_keys([1, 2, 3, 4])*

**values()-**returns all the keys in the dictionary

print("return all the values")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print(dictOne.values()) *#dict\_values(['Horse', 'Dog', 'Cat', 'Lion'])*

**items()—**returns each item in dictionary as tuples in a list. meaning that any changes done to the dictionary will be reflected in the items list.

print("Make a change in the original dictionary, and see that the items list gets updated as wel")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
listOne=dictOne.items()  
print(listOne)*# dict\_items([(1, 'Horse'), (2, 'Dog'), (3, 'Cat'), (4, 'Lion')])*dictOne[5]='Bunny' # adding a new item  
print(dictOne)*# {1: 'Horse', 2: 'Dog', 3: 'Cat', 4: 'Lion', 5: 'Bunny'}*print(listOne)*#dict\_items([(1, 'Horse'), (2, 'Dog'), (3, 'Cat'), (4, 'Lion'), (5, 'Bunny')])*

**Change the values**

**Referring the key name:**

print("Change the \"year\" to 2018")  
thisdict = {  
 "brand": "Ford",  
 "model": "Mustang",  
 "year": 1964  
}  
thisdict["year"]=2018  
print(thisdict) *#{'brand': 'Ford', 'model': 'Mustang', 'year': 2018}*

**Update()—**update the dictionary with the items from the given argument. The argument must be a dictionary, or an iterable object with key:value pairs.

print("Update the \"year\" of the car by using the update() method")  
thisdict = {  
 "brand": "Ford",  
 "model": "Mustang",  
 "year": 1964  
}  
thisdict.update({"year":2018})  
print(thisdict) *#{'brand': 'Ford', 'model': 'Mustang', 'year': 2018}*

**Removing Items**

**Pop()** removes the item with the specified key name

print("Remove item of specified value using pop()")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
dictOne.pop(1)  
print(dictOne)*#{2: 'Dog', 3: 'Cat', 4: 'Lion'}*

**popitem()** method removes the last inserted item (in versions before 3.7, a random item is removed instead):

print("Remove the last item using popitem()")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
dictOne.popitem()  
print(dictOne)*#{1: 'Horse', 2: 'Dog', 3: 'Cat'}*

**del** keyword removes the item with the specified key name or can also delete the dictionary completely

print("Remove the specified item using del()")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
del dictOne[4]  
print(dictOne)*#{1: 'Horse', 2: 'Dog', 3: 'Cat'}*

**delete the dictionary**

print("Remove dictionary using del")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
del dictOne  
print(dictOne)*#NameError: name 'dictOne' is not defined*

**clear()** method empties the dictionary:

print("clear the items in dictionary using clear")  
dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
dictOne.clear()  
print(dictOne)*#{}*

**Looping through a dictionary**, the return value are the **keys** of the dictionary, but there are methods to return the **values** as well.

dictOne = {  
 1:'Horse',  
 2:'Dog',  
 3:'Cat',  
 4:'Lion'  
}  
print("Print all key names in the dictionary one by one")  
for i in dictOne:  
 print(i)  
*#1  
#2  
#3  
#4*print("Print all values in the dictionary one by one")  
for i in dictOne:  
 print(dictOne[i])  
*#Horse  
#Dog  
#Cat  
#Lion*print("return values from the dictionary")  
for i in dictOne.values():  
 print(i)  
*#Horse  
#Dog  
#Cat  
#Lion*

print("Print all keys in the dictionary one by one")  
for i in dictOne.keys():  
 print(i)  
*#1  
#2  
#3  
#4*

print("Loop through both keys and values, by using the items() method")  
for i in dictOne.items():  
 print(i)  
*#(1, 'Horse')  
#(2, 'Dog')  
#(3, 'Cat')  
#(4, 'Lion')*

**Copy Dictionaries**

**Copy()-**

print("Make a copy of a dictionary with the copy() method")  
dictOne = {1:'Horse', 2:'Dog', 3:'Cat', 4:'Lion'}  
dictTwo = dictOne.copy()  
print(dictTwo) *#{1: 'Horse', 2: 'Dog', 3: 'Cat', 4: 'Lion'}*

**dict()-**

print("Make a copy of a dictionary with the dict() function")  
dictOne = {1:'Horse', 2:'Dog', 3:'Cat', 4:'Lion'}  
dictTwo = dict(dictOne)  
print(dictTwo) *#{1: 'Horse', 2: 'Dog', 3: 'Cat', 4: 'Lion'}*

**Nested Dictionary**

A dictionary can contain sub dictionaries.

print("Create a dictionary that contain three dictionaries")  
animal = {  
 "an\_1":{1:'Horse', 2:'Dog'},  
 "an\_2":{3:'Cat', 4:'Lion'},  
 "an\_3":{5:"Snake", 6:"Elephant"}  
}  
print(animal)  
*#{'an\_1': {1: 'Horse', 2: 'Dog'}, 'an\_2': {3: 'Cat', 4: 'Lion'}, 'an\_3': {5: 'Snake', 6: 'Elephant'}}*

print("Create three dictionaries, then create one dictionary by adding them")  
an\_1 = {1:'Horse', 2:'Dog'}  
an\_2 = {3:'Cat', 4:'Lion'},  
an\_3 = {5:"Snake", 6:"Elephant"}  
  
animal = {  
 "an\_1": an\_1,  
 "an\_2": an\_2,  
 "an\_3": an\_3  
}  
print(animal)*#{'an\_1': {1: 'Horse', 2: 'Dog'}, 'an\_2': ({3: 'Cat', 4: 'Lion'},), 'an\_3': {5: 'Snake', 6: 'Elephant'}}*

fromkeys()-create a dictionary with a specified keys and values

print("create a dictionary with a specified keys and values")  
k1 = (1, 2, 3)  
v1 = "x"  
myDict = dict.fromkeys(k1, v1)  
print(myDict)*#{1: 'x', 2: 'x', 3: 'x'}*

**setdefault(**keyname(required),value(optional**))** method returns the value of the item with the specified key. If the key does not exist, insert the key, with the specified value otherwise value will be none

**Get the value of the \"model\" item when the key-value exists**  
car = {  
 "brand": "Ford",  
 "model": "Mustang",  
 "year": 1964  
}  
x=car.setdefault("model","Bronco")  
print(x) *#Mustang*print(car)*#{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}*

**set default item with none when the key does not exist**  
car = {  
 "brand": "Ford",  
 "year": 1964  
}  
x=car.setdefault("model")  
print(x) *#Mustang*print(car)*#{'brand': 'Ford', 'year': 1964, 'model': None}*

**set default item with value when the key does not exist"**car = {  
 "brand": "Ford",  
 "year": 1964  
}  
x=car.setdefault("model","Bronco")  
print(x) *#Mustang*print(car)*#{'brand': 'Ford', 'year': 1964, 'model': 'Bronco'}*

Dictionary methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [clear()](https://www.w3schools.com/python/ref_dictionary_clear.asp) | Removes all the elements from the dictionary |
| [copy()](https://www.w3schools.com/python/ref_dictionary_copy.asp) | Returns a copy of the dictionary |
| [fromkeys()](https://www.w3schools.com/python/ref_dictionary_fromkeys.asp) | Returns a dictionary with the specified keys and value |
| [get()](https://www.w3schools.com/python/ref_dictionary_get.asp) | Returns the value of the specified key |
| [items()](https://www.w3schools.com/python/ref_dictionary_items.asp) | Returns a list containing a tuple for each key value pair |
| [keys()](https://www.w3schools.com/python/ref_dictionary_keys.asp) | Returns a list containing the dictionary's keys |
| [pop()](https://www.w3schools.com/python/ref_dictionary_pop.asp) | Removes the element with the specified key |
| [popitem()](https://www.w3schools.com/python/ref_dictionary_popitem.asp) | Removes the last inserted key-value pair |
| [setdefault()](https://www.w3schools.com/python/ref_dictionary_setdefault.asp) | Returns the value of the specified key. If the key does not exist: insert the key, with the specified value |
| [update()](https://www.w3schools.com/python/ref_dictionary_update.asp) | Updates the dictionary with the specified key-value pairs |
| [values()](https://www.w3schools.com/python/ref_dictionary_values.asp) | Returns a list of all the values in the dictionary |