**Dictionary Functions and Methods**

# Functions

|  |  |  |
| --- | --- | --- |
| S. No. | Function | Description |
| 1 | **cmp(dict1, dict2)** | No longer available in Python 3. |
| 2 | **len(dict)** | Gives the total number of items in the dictionary. |
| 3 | **str(dict)** | Produces a printable string representation of a dictionary |
| 4 | **type(variable)** | Returns the type of the passed variable. |

dict = {'name' : 'My Name', 'age': 22, 'job': 'none'}

print("len(dict): ", len(dict))

print("str(dict): ", str(dict))

print("type(dict): ", type(dict))

Output:

len(dict): 3

str(dict): {'age': 22, 'name': 'My Name', 'job': 'none'}

type(dict): &lt;class 'dict'&gt;

# Methods

Call following methods on dictionary object in following way:

dic.method\_name();

|  |  |  |
| --- | --- | --- |
| S. No. | Method | Description |
| 1 | **clear()** | Removes all elements of dictionary *dict* |
| 2 | **copy()** | Returns a shallow copy of dictionary *dict* |
| 3 | **fromkeys()** | Create a new dictionary with keys from seq and values *set* to *value*. |
| 4 | **get(key, default=None)** | For *key* key, returns value or default if key not in dictionary |
| 5 | **has\_key(key)** | Removed, use the *in* operation instead. |
| 6 | **items()** | Returns a list of *dict*'s (key, value) tuple pairs |
| 7 | **keys()** | Returns list of dictionary dict's keys |
| 8 | **setdefault(key, default = None)** | Similar to get(), but will set dict[key] = default if *key* is not already in dict |
| 9 | **update(dict2)** | Adds dictionary *dict2*'s key-values pairs to *dict* |
| 10 | **values()** | Returns list of dictionary *dict*'s values |

## copy

dict2 = {'name': 'Myname', 'age': 7, 'job': 'none'}

dict1 = dict2.copy()

print("dict1['name']: ", dict1['name'])

print("dict1['age']: ", dict1['age'])

print("dict1['job']: ", dict1['job'])

print("dict2['job']: ", dict2['job'])

del dict2['job']

print("dict1['name']: ", dict1['name'])

print("dict1['age']: ", dict1['age'])

print("dict1['job']: ", dict1['job']) # ????

#print("dict2['job']: ", dict2['job']) # KeyError: 'job'

## fromkeys

creates a new dictionary with keys from seq and values set to value

**Syntax**

dict.fromkeys(seq[, value]))

seq = ('name', 'age', 'sex')

dict = dict.fromkeys(seq)

print ("New Dictionary : %s" % str(dict))

dict = dict.fromkeys(seq, 10)

print ("New Dictionary : %s" % str(dict))

Output:

New Dictionary : {'name': None, 'sex': None, 'age': None}

New Dictionary : {'name': 10, 'sex': 10, 'age': 10}

## get

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

print("dict.get('name', 0): ", dict.get('name', 0))

print("dict.get('na', 0): ", dict.get('na', 0))

Output:

dict.get('name', 0): Myname

dict.get('na', 0): 0

## has\_key

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

#print("dict.has\_key('name'): ", dict.has\_key('name')) # AttributeError: 'dict' object has no attribute 'has\_key'

## items

returns a list of dict's (key, value) tuple pairs

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

print("dict.items(): ", dict.items())

Output:

dict.items(): dict\_items([('job', 'none'), ('name', 'Myname'), ('age', 7)])

## keys

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

print("dict.keys(): ", dict.keys())

Output:

dict.keys(): dict\_keys(['job', 'age', 'name'])

## values

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

print("dict.values(): ", dict.values())

Output:

dict.values(): dict\_values(['Myname', 'none', 7])

## setdefault()

similar to get(), but will set dict[key] = default if key is not already in dict.

dict = {'name': 'Myname', 'age': 7, 'job': 'none'}

print(dict)

print("dict.setdefault('name', 0): ", dict.setdefault('name', 0))

print("dict.setdefault('group', 'NA'): ", dict.setdefault('group', 'NA'))

print(dict)

Output:

{'job': 'none', 'name': 'Myname', 'age': 7}

dict.setdefault('name', 0): Myname

dict.setdefault('group', 'NA'): NA

{'job': 'none', 'name': 'Myname', 'age': 7, 'group': 'NA'}

## update()

adds dictionary dict2's key-values pairs in to dict. This function does not return anything

dict1 = {'name': 'Myname', 'age': 7, 'job': 'none'}

dict2 = {'group': 'NA'}

dict = dict1.update(dict2)

print(dict1)

print(dict2)

print(dict)

Output:

{'group': 'NA', 'name': 'Myname', 'age': 7, 'job': 'none'}

{'group': 'NA'}

None