Dictionaries

A dictionary is a collection of unordered, modifiable(mutable) paired (key: value) data type.

Creating a Dictionary

To create a dictionary we use curly brackets, {} or the *dict()* built-in function.

```
# syntax
empty_dict = {}
# Dictionary with data values
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'6th main street',
        'zipcode':'560037'
    }
    }
```

The dictionary above shows that a value could be any data types:string, boolean, list, tuple, set or a dictionary.

Dictionary Length

It checks the number of 'key: value' pairs in the dictionary.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(len(dct)) # 4
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':35,
    'country':'India',
    'is_married':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'6th Main street',
        'zipcode':'560037'
    }
    }
print(len(person)) # 7
```

Accessing Dictionary Items

We can access Dictionary items by referring to its key name.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct['key1']) # value1
print(dct['key4']) # value4
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
print(person['first_name']) # Prasanta
print(person['country'])    # India
print(person['skills'])     # ['JavaScript', 'React', 'Node', 'MongoDB',
'Python']
print(person['skills'][0])  # JavaScript
print(person['address']['street']) # Space street
print(person['city'])       # Error
```

Accessing an item by key name raises an error if the key does not exist. To avoid this error first we have to check if a key exist or we can use the *get* method. The get method returns None, which is a NoneType object data type, if the key does not exist.

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
print(person.get('first_name')) # Prasanta
print(person.get('country'))    # India
print(person.get('skills')) #['HTML','CSS','JavaScript', 'React', 'Node',
'MongoDB', 'Python']
print(person.get('city'))   # None
```

Adding Items to a Dictionary

We can add new key and value pairs to a dictionary

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct['key5'] = 'value5'
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'6th Main street',
        'zipcode':'560037'
        }
}
person['job_title'] = 'Instructor'
person['skills'].append('HTML')
print(person)
```

Modifying Items in a Dictionary

We can modify items in a dictionary

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct['key1'] = 'value-one'
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'6th Main street',
        'zipcode':'560037'
    }
    }
person['first_name'] = 'Eyob'
person['age'] = 252
```

Checking Keys in a Dictionary

We use the *in* operator to check if a key exist in a dictionary

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print('key2' in dct) # True
print('key5' in dct) # False
```

Removing Key and Value Pairs from a Dictionary

- ***pop(key)***: removes the item with the specified key name:
- ***popitem()***: removes the last item
- ***del***: removes an item with specified key name

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct.pop('key1') # removes key1 item
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct.popitem() # removes the last item
del dct['key2'] # removes key2 item
```

Example:

```
person = {
    'first_name':'Prasanta',
    'last_name':'Biswal',
    'age':250,
    'country':'India',
    'is_marred':True,
    'skills':['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address':{
        'street':'Space street',
        'zipcode':'02210'
    }
    }
person.pop('first_name')        # Removes the firstname item
person.popitem()                # Removes the address item
del person['is_married']        # Removes the is_married item
```

Changing Dictionary to a List of Items

The *items()* method changes dictionary to a list of tuples.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct.items()) # dict_items([('key1', 'value1'), ('key2', 'value2'),
('key3', 'value3'), ('key4', 'value4')])
```

Clearing a Dictionary

If we don't want the items in a dictionary we can clear them using *clear()* method

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
print(dct.clear()) # None
```

Deleting a Dictionary

If we do not use the dictionary we can delete it completely

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
del dct
```

Copy a Dictionary

We can copy a dictionary using a *copy()* method. Using copy we can avoid mutation of the original dictionary.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
dct_copy = dct.copy() # {'key1':'value1', 'key2':'value2', 'key3':'value3',
'key4':'value4'}
```

Getting Dictionary Keys as a List

The ***keys()*** method gives us all the keys of a a dictionary as a list.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
keys = dct.keys()
print(keys)     # dict_keys(['key1', 'key2', 'key3', 'key4'])
```

Getting Dictionary Values as a List

The *values* method gives us all the values of a a dictionary as a list.

```
# syntax
dct = {'key1':'value1', 'key2':'value2', 'key3':'value3', 'key4':'value4'}
values = dct.values()
print(values)     # dict_values(['value1', 'value2', 'value3', 'value4'])
```