Dictionary in Python is like an object containing multiple key value pairs.

Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value. Following is a simple example −

```
dict = {'Name': 'Krish', 'Age': 35}
print "dict['Name']: ", dict['Name']
print "dict['Age']: ", dict['Age']
```

When the above code is executed, it produces the following result −

```
dict['Name']:  Krish
dict['Age']:  35
```

If we attempt to access a data item with a key, which is not a part of the dictionary,then Python will throw error as follows −

```
dict = {'Name': 'Krish', 'Age': 35}
print "dict['Car']: ", dict['Car']
```

When the above code is executed, it produces the following result −

```
dict['Car']:
Traceback (most recent call last):
   File "test.py", line 2, in <module>
      print "dict['Car']: ", dict['Car'];
KeyError: 'Car'
```

Updating Dictionary

You can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry as shown below in the simple example −

```
dict = {'Name': 'Krish', 'Age': 35, 'Car': 'Audi'}
dict['Car'] =  ‘bmw’  # update existing entry
dict['Country'] = "US" # Add new entry
print "dict['Age']: ", dict['Age']
print "dict['Country']: ", dict['Country']
```

When the above code is executed, it produces the following result −

```
dict['Age']:  35
dict['Country']:  US
```

Delete Dictionary Elements

You can use del statement either to remove individual dictionary elements or clear the entire contents of a dictionary.

You can also delete entire dictionary in a single operation.

Following is a simple example −

```
dict = {'Name': 'Krish', 'Age': 35, 'Car': 'Audi'}
del dict['Name'] # remove entry with key 'Name'
dict.clear()     # remove all entries in dict
# delete entire dictionary
print "dict['Age']: ", dict['Age']
print "dict['Car']: ", dict['Car']
```

This produces the following result. Note that an exception is raised because after **del dict** dictionary does not exist any more −

```
dict['Age']:
Traceback (most recent call last):
   File "test.py", line 5, in <module>
      print "dict['Age']: ", dict['Age'];
TypeError: 'type' object is unsubscriptable
```