# Python - Hash Table

Hash tables are a type of data structure in which the address or the index value of the data element is generated from a hash function. That makes accessing the data faster as the index value behaves as a key for the data value. In other words Hash table stores key-value pairs but the key is generated through a hashing function.

So the search and insertion function of a data element becomes much faster as the key values themselves become the index of the array which stores the data.

In Python, the Dictionary data types represent the implementation of hash tables. The Keys in the dictionary satisfy the following requirements.

- The keys of the dictionary are hashable i.e. the are generated by hashing function which generates unique result for each unique value supplied to the hash function.
- The order of data elements in a dictionary is not fixed.

So we see the implementation of hash table by using the dictionary data types as below.

## Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value.

### Example

```
# Declare a dictionary
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

# Accessing the dictionary with its key
print "dict['Name']: ", dict['Name']
print "dict['Age']: ", dict['Age']
```

### Output

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

```
dict['Name']:  Zara
dict['Age']:  7
```

## 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 −

### Example

```
# Declare a dictionary
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
dict['Age'] = 8; # update existing entry
dict['School'] = "DPS School"; # Add new entry
print "dict['Age']: ", dict['Age']
print "dict['School']: ", dict['School']
```

### Output

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

```
dict['Age']:  8
dict['School']:  DPS School
```

## Delete Dictionary Elements

You can either remove individual dictionary elements or clear the entire contents of a dictionary. You can also delete entire dictionary in a single operation.To explicitly remove an entire dictionary, just use the del statement.

### Example

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
del dict['Name']; # remove entry with key 'Name'
dict.clear();     # remove all entries in dict
del dict ;        # delete entire dictionary

print "dict['Age']: ", dict['Age']
print "dict['School']: ", dict['School']
```

### Output

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

```
dict['Age']:
Traceback (most recent call last):
   File "test.py", line 8, in <module>
      print "dict['Age']: ", dict['Age'];
TypeError: 'type' object is unsubscriptable
```

**Print Page**