DICTIONARY

Module 5

Module Outline:

- Dictionary Overview
- Dictionary Representation in Python
- Dictionary Basic Operations
    - Accessing
    - Insertion
    - Deletion
    - Search
    - Update

What is a Dictionary?

Overview

- In Dictionary each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces.
  - An empty dictionary without any items is written with just two curly braces, like this: {}.
- 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.

Dictionary Operations

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': 'Zara', 'Age': 7, 'Class': 'First'}
print ("dict['Name']: ", dict['Name’])
print ("dict['Age']: ", dict['Age’])
```

Accessing Values in Dictionary

If we attempt to access a data item with a key, which is not part of the dictionary, we get an **error** as follows –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
print "dict[Juan']: ", dict['Juan’]
```

Error:

```
dict[Juan’]:
Traceback (most recent call last):
        File "test.py", line 4, in <module>
                print "dict[Juan ']: ", dict[Juan’];
KeyError: Juan '
```

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 –

Updating 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']
```

What is the output?

Deleting 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. Following is a simple example –

Deleting List Elements

```
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']
```

What is the output?

Properties of Dictionary Keys

Properties of Dictionary Keys

- Dictionary values have **no restrictions**. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.
- There are two important points to remember about dictionary keys –

Properties of Dictionary Keys

- **(a) More than one entry per key not allowed**. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins. For example –

```
dict = {'Name': 'Zara', 'Age': 7, 'Name': 'Manni'}
print "dict['Name']: ", dict['Name’]
```

What is the output?

Properties of Dictionary Keys

- **(b)** Keys must be **immutable** and **‘unhashable’.** Which means you can use strings, numbers or tuples as dictionary keys but something like ['key'] is not allowed. Following is a simple example –

```
dict = {['Name']: 'Zara', 'Age': 7}
print "dict['Name']: ", dict['Name’]
```

What is the output?

* End of Module *

Reference:

The instructor does not take the credits on the contents of this presentation.

https://www.tutorialspoint.com/python_data_structure/