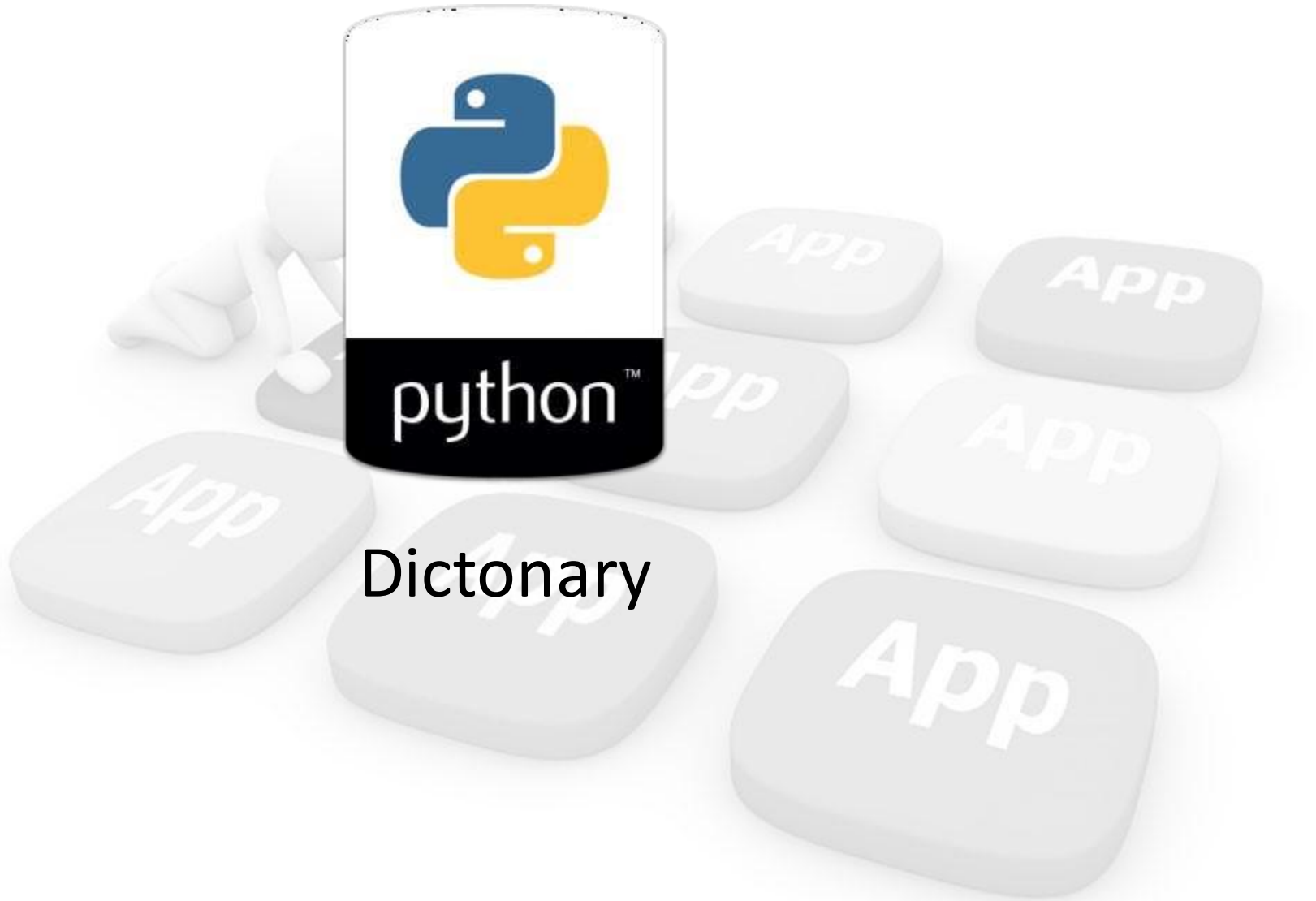
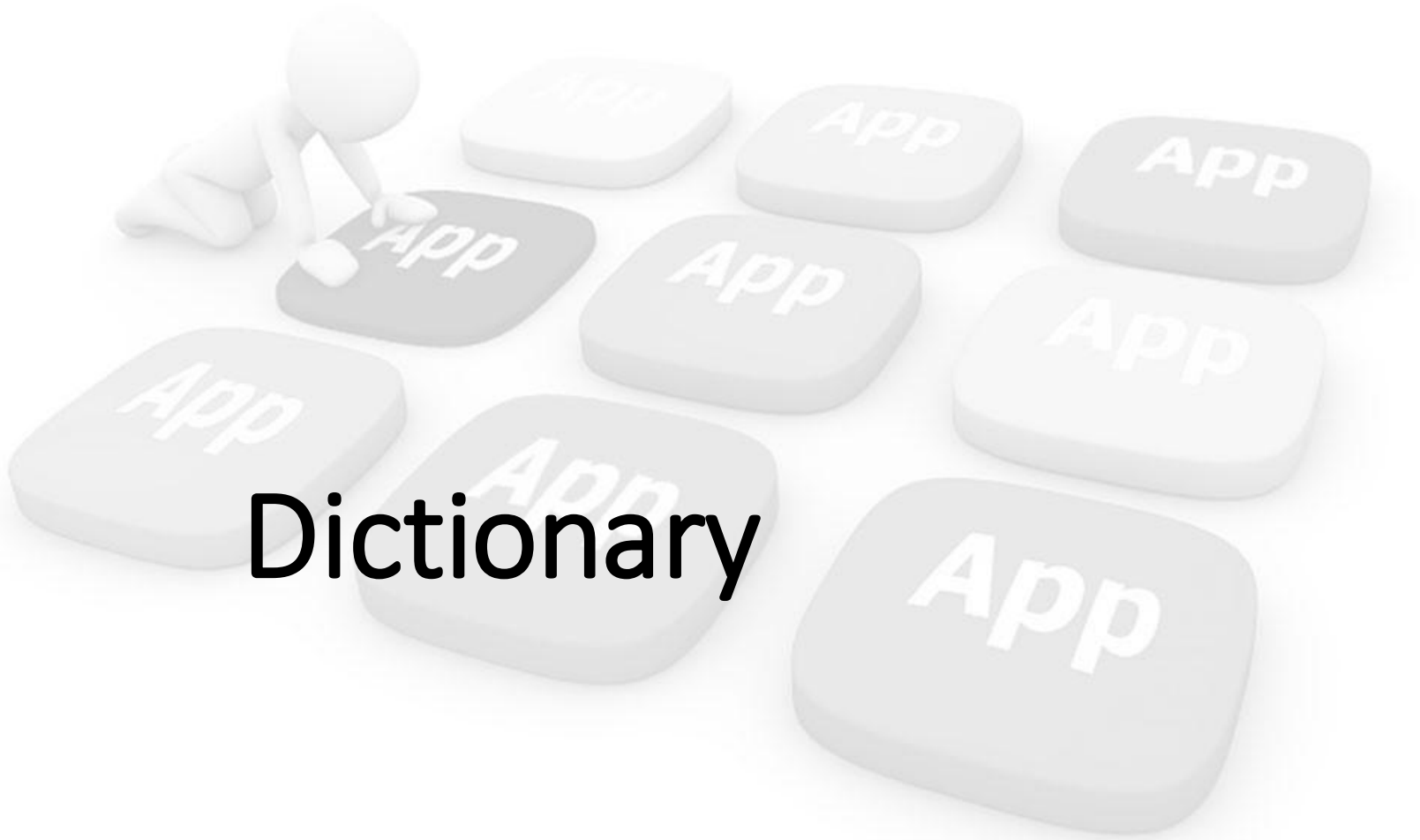


Dictonary



Dictionary

Introduction to Dictionary

- Dictionaries are another example of a data structure. A dictionary is used to map or associate things you want to store the keys you need to get them.
- A dictionary in Python is just like a dictionary in the real world. Python Dictionary are defined into two elements Keys and Values.
- Keys will be a single element
- Values can be a list or list within a list, numbers, etc
- **Syntax for Python Dictionary**:
- Dict={‘Tim’:10, xyz,…}
- Dictionary is listed in curly brackets, inside these curly brackets, keys and values are declared. Each key is separated from its value by a colon (:) while each element is separated by commas.

Properties of Dictionary Keys

- There are two important points while using dictionary keys
- More than one entry per key is not allowed (no duplicate key is allowed)
- The values in the dictionary can be of any type while the keys must be immutable like numbers, tuples or strings.
- Dictionary keys are case sensitive- Same key name but with the different case are treated as different keys in Python dictionaries.

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'])
```

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 –

```
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'])
```

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. Following is a simple 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'])
```

Delete Dictionary Elements(cont)

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 8, in <module>
    print ("dict['Age']: ", dict['Age'])
TypeError: 'type' object is unsubscriptable
```

Function	Description
cmp()	Compares elements of both dict. cmp(dict1,dict2). This method returns 0 if both dictionaries are equal, -1 if dict1 < dict2 and 1 if dict1 > dic2
keys()	Returns list of dictionary dict's keys
pop()	Removes and returns an element from a dictionary having the given key.
clear()	The clear() method removes all items from the dictionary.

Thank You