Rakesh Raja KB, RA1931005020188
BSc. Cs D sec.

Dictionaries :

- A Python dictionary is written as a sequence of key/value pairs separated by commas. These pairs are called entries.

- The entire sequence of entries is enclosed in curly braces ({and}, A colon (:) separates a key and its value.

Examples
phonebook: {"udaya": "9791113940", "Gowtham": "9003273969"}
personal - information: {"Name": "Kamesh", "Age": 20}

Adding keys and Replacing Values

- Add a new key/value pair to a dictionary by using the subscript operator [] <a dictionary>[<a keys>]=<a value>

To create a new empty dictionary and add two new entries

```
info = {}
info["name"] = "Sandy"
info["occupation"] = "Manager"
info
{'name': 'sandy', 'occupation': 'Manager'}
```

The subscript is also used to replace a value of an existing key.

```
info ["occupation"] = "Developer"
info
{'name': 'Sandy', 'occupation': 'Developer'}
```

Accessing Values

- The subscript operator [] is used to obtain a value associated with a key.

- However, if the key is not present in the dictionary, Python raises an exception.

```
info["name"]
'Sandy'
```

```
info["job"]
-------------------------------------------------
KeyError                 Traceback (most recent call last)
<ipython-input-6-fbe3dde50412>in <module>()
---> 1 info["job"]

KeyError: 'job'

[Search Stack Overflow]
```

Removing Keys

- To delete an entry from a dictionary, one removes its key using the method pop.
- This method expects a key and an optional ~~defflow~~ default value as arguments.
- If the key is in the dictionary, it is removed, and its associated value is returned. Otherwise, the default value is returned.

```
[7] print (info.pop("job", None))
    None
    print (info. pop ("occupation"))
    Developer
```

Traversing a Dictionary

- When a for loop is used with a dictionary, the loop's variable is bound to each key in an unspecified order.
- The code segment prints all of the keys and their values.

```
▷ for key in info:
      print(key, info[key])

↦ name Sandy
```

```
▷ grades = {90:'A', 80:'B', 70:'C'}
  list (grades.items())

[(90,'A'), (80, 'B'), (70,'C')]
```

Ⓓ
```
for (key, value) in grades.items():
print (key, value)
90  A
80  B
70  C
```

<u>Dictionary Operations</u>

Operator or Function	What it does
len (d)	Returns the number of entries in d.
d [key]	Used for inserting a new key, replacing a value or obtaining a value at an existing key.
d. clear	Removes all the keys
list (d. keys ())	Returns a list of the keys
list (d. values ())	Returns a list of the values.
list (d. items ())	Returns a list of tuples containing the keys and values for each entry
For key in d:	key is bound to each key in d in an unspecified order.

```
this dict = {
 "name": "Rakesh",
"Native" : "Chennai",
"year"   : 2001
}
```

```
#thisdict
print("1. print the dictionary:\n", thisdict)
#len()
print("\n2. return the number of entries in thisdict:", len(thisdict))
#keys()
print("\n3. Return the list of the keys", thisdict.keys())
#values()
print("\n4. Return the list of the value", thisdict.values())
#items
print("\n5. Return the list of the tuples containing keys and values",
                                              thisdict.items())
#copy()
print("\n6. Returns a copy of the dictionary", thisdict.copy())
#pop()
print("\n7. Removes the element with the specified key:", thisdict.pop
                                                          ("year"))

#get()
print("\n8. Removes the element with the specified key:", thisdict.get
                                                       ("name"))

#setdefault
print("\n9. Returns the value of the specified key:", thisdict.setdefault
                                                   ("name"))

#popitem
print("\n10. The removed item is the return value of the
         popitem() method, as a tuple:", thisdict.popitem())
#update
thisdict.update({"color": "Black"}}
print("\n11.", thisdict)
#clear
print("\n12. Remove all keys", thisdict.clear())
```