7. Dictionary

October 18, 2022

1 Introduction

- Python dictionary is an unordered collection of items.
- Each item of a dictionary has a key/value pair.
- Dictionaries are optimized to retrieve values when the key is known.
- Creating a dictionary is as simple as placing items inside curly braces `{}` separated by commas.
- An item has a key and a corresponding value that is expressed as a pair`key: value`.

```
[1]: city_temp =  {'Mumbai': 32, 'Hyderabad': 38,
                   'Vishakhapatnam': 31, 'Delhi': 42}
```

```
[2]: city_temp
```

```
[2]: {'Mumbai': 32, 'Hyderabad': 38, 'Vishakhapatnam': 31, 'Delhi': 42}
```

```
[3]: type(city_temp)
```

```
[3]: dict
```

```
[4]: len(city_temp)
```

```
[4]: 4
```

2 Properties of Dictionary

- Dictionaries are mutable, so we can easily add or remove items
- Unordered
- Have no notion of index position and so cannot be sliced
- Keys
  - No duplicate keys allowed
  - Keys must be immutable – strings, numbers, tuples
- Values: have no restrictions – Any object, built-in type or user defined – Numbers, strings, lists, sets, dictionaries etc.

3 Creating Dictionary using `dict()` function

```
[5]: d1 = dict([('two', 2), ('one', 1), ('three', 3)])
```

```
[6]: d1
```

```
[6]: {'two': 2, 'one': 1, 'three': 3}
```

4 Accessing Values from dictionary

- While indexing is used with other data types to access values, a dictionary uses keys. Keys can be used either inside square brackets `[]` or with the `get()` method.
- If we use the square brackets `[]`, `KeyError` is raised in case a key is not found in the dictionary. On the other hand, the `get()` method returns `None` if the key is not found.

```
[7]: city_temp
```

```
[7]: {'Mumbai': 32, 'Hyderabad': 38, 'Vishakhapatnam': 31, 'Delhi': 42}
```

```
[8]: city_temp['Mumbai']
```

```
[8]: 32
```

```
[9]: city_temp['Vishakhapatnam']
```

```
[9]: 31
```

```
[10]: city_temp['Pune']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In [10], line 1
----> 1 city_temp['Pune']

KeyError: 'Pune'
```

5 Modifying a dictionary

5.1 Changing available values

```
[11]: city_temp
```

```
[11]: {'Mumbai': 32, 'Hyderabad': 38, 'Vishakhapatnam': 31, 'Delhi': 42}
```

```
[12]: city_temp['Hyderabad'] = 40
```

```
[13]: city_temp
```

```
[13]: {'Mumbai': 32, 'Hyderabad': 40, 'Vishakhapatnam': 31, 'Delhi': 42}
```

5.2 Inserting new key-value pairs

```
[14]: city_temp
```

```
[14]: {'Mumbai': 32, 'Hyderabad': 40, 'Vishakhapatnam': 31, 'Delhi': 42}
```

```
[15]: city_temp['Bhopal'] = 38
```

```
[16]: city_temp
```

```
[16]: {'Mumbai': 32,
       'Hyderabad': 40,
       'Vishakhapatnam': 31,
       'Delhi': 42,
       'Bhopal': 38}
```

5.3 Deleting existing key-value pairs

```
[17]: del city_temp['Vishakhapatnam']
```

```
[18]: city_temp
```

```
[18]: {'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}
```

```
[19]: del city_temp # Delete whole Dictionary
```

```
[20]: city_temp
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In [20], line 1
----> 1 city_temp

NameError: name 'city_temp' is not defined
```

6 Methods of Dictionary

1. keys()
2. values()
3. items()
4. get()

5. str()
6. pop()
7. popitem()
8. update()
9. clear()

[21]:
```
city_temp = {'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}
```

6.1 keys()

- Return all the keys in a dictionary

[22]:
```
city_temp.keys()
```

[22]:
```
dict_keys(['Mumbai', 'Hyderabad', 'Delhi', 'Bhopal'])
```

6.2 values()

- Return all the values in a dictionary

[23]:
```
city_temp.values()
```

[23]:
```
dict_values([32, 40, 42, 38])
```

6.3 items()

[24]:
```
city_temp.items()
```

[24]:
```
dict_items([('Mumbai', 32), ('Hyderabad', 40), ('Delhi', 42), ('Bhopal', 38)])
```

6.4 get()

- Returns value of given key if key is in the dictionary, else default.

[25]:
```
city_temp.get("Bhopal")
```

[25]:
```
38
```

[26]:
```
city_temp["Bhopal"] # Using square brackets to get values from dict
```

[26]:
```
38
```

[28]:
```
city_temp
```

[28]:
```
{'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}
```

[27]:
```
city_temp.get("Pune")
```

[29]:
```
type(city_temp.get("Pune"))
```

[29]:
```
NoneType
```

[30]:
```
city_temp['pune']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In [30], line 1
----> 1 city_temp['pune']

KeyError: 'pune'
```

[31]:
```
city_temp.get("Pune", 24)
```

[31]:
```
24
```

[32]:
```
city_temp
```

[32]:
```
{'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}
```

[33]:
```
city_temp.get("Mumbai",40)
```

[33]:
```
32
```

6.5 str()

[34]:
```
city_temp
```

[34]:
```
{'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}
```

[35]:
```
str(city_temp)
```

[35]:
```
"{'Mumbai': 32, 'Hyderabad': 40, 'Delhi': 42, 'Bhopal': 38}"
```

6.6 pop()

- Remove specified key and return the corresponding value.
- If key is not found, default is returned if given, otherwise KeyError is raised

[36]:
```
city_temp.pop("Delhi")
```

[36]:
```
42
```

[37]:
```
city_temp
```

```
[37]: {'Mumbai': 32, 'Hyderabad': 40, 'Bhopal': 38}
```

```
[38]: city_temp.pop("Pune")
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In [38], line 1
----> 1 city_temp.pop("Pune")

KeyError: 'Pune'
```

```
[39]: city_temp.pop("Pune", 38)  # default parameter same as get() method
```

```
[39]: 38
```

6.7 popitem()

- Remove and return a (key, value) pair as a 2-tuple.
- Pairs are returned in LIFO (last-in, first-out) order.
- Raises KeyError if the dict is empty.

```
[40]: city_temp
```

```
[40]: {'Mumbai': 32, 'Hyderabad': 40, 'Bhopal': 38}
```

```
[41]: city_temp.popitem()
```

```
[41]: ('Bhopal', 38)
```

```
[42]: city_temp
```

```
[42]: {'Mumbai': 32, 'Hyderabad': 40}
```

6.8 update(other)

- Updates the dictionary with the key/value pairs from other, overwriting existing keys.

```
[43]: city_temp
```

```
[43]: {'Mumbai': 32, 'Hyderabad': 40}
```

```
[44]: ct2 = {'Bangalore': 30, 'Mumbai': 36}
```

```
[45]: city_temp.update(ct2)
```

```
[46]: city_temp
```

```
[46]: {'Mumbai': 36, 'Hyderabad': 40, 'Bangalore': 30}
```

```
[47]: ct2
```

```
[47]: {'Bangalore': 30, 'Mumbai': 36}
```

6.9 clear()

- Remove all items from Dictionary

```
[48]: city_temp.clear()
```

```
[49]: city_temp
```

```
[49]: {}
```

7 For Loop on Dictionary

```
[50]: personal_info = {"username": "spidey",
                       "email": "spidey@starkindustries.com",
                       "location":"New York City",
                       "firstName" : "Peter",
                       "lastName": "Parker"}
```

```
[51]: for i in personal_info:
          print(i)
          print("_"*5)
```

```
username
_____
email
_____
location
_____
firstName
_____
lastName
_____
```

```
[52]: for i in personal_info:
          print(personal_info[i])
          print("_"*5)
```

```
spidey
_____
spidey@starkindustries.com
_____
```

```
New York City
_____
Peter
_____
Parker
_____
```

[53]:
```
for i in personal_info:
    print(i,':',personal_info[i])
    print("_"*5)
```

```
username : spidey
_____
email : spidey@starkindustries.com
_____
location : New York City
_____
firstName : Peter
_____
lastName : Parker
_____
```

[54]:
```
for i in personal_info.keys():
    print(i)
    print("_"*5)
```

```
username
_____
email
_____
location
_____
firstName
_____
lastName
_____
```

[56]:
```
for i in personal_info.values():
    print(i)
    print("_"*5)
```

```
('username', 'spidey')
_____
('email', 'spidey@starkindustries.com')
_____
('location', 'New York City')
_____
('firstName', 'Peter')
```

```
-----
('lastName', 'Parker')

-----
```