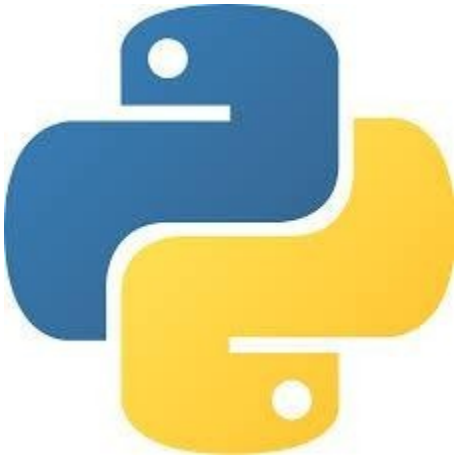
Dictionary in Python



Dictionary Creation

Dictionary in Python

- In python, a dictionary can be defined as a collection of key-values pairs.
- Each key is unique, and is separated from its value by a colon (:).
- Key-value pairs are enclosed in curly braces {}.

Syntax:

dict_var = { key1: value1, key2: value2, … }

Example: "dictdemo.py"

```
D1 = {}
D2 = {1: "apple", 2: "banana", 3: "cherry"}
D3 = {"name": "John", "age": 25, "city": "Delhi"}
print(D1)
print(D2)
print(D3)
```

Output

```
{}
{1: 'apple', 2: 'banana', 3: 'cherry'}
{'name': 'John', 'age': 25, 'city': 'Delhi'}
```

Dictionary Indexing

Dictionary Indexing in Python

- Dictionary values are accessed using keys, not indexes.
- Use square brackets [] or get() method.

```
mydict = {"name": "Alice", "age": 22, "city": "London"}
print(mydict["name"])   # Alice
print(mydict.get("age")) # 22
```

Updating Dictionary in Python

- You can change values by assigning a new one to an existing key.
- Add new key-value pairs simply by assignment.

Example

```
student = {"name": "Bob", "age": 20}
student["age"] = 21
student["branch"] = "CSE"
print(student)
```

Output

```
{'name': 'Bob', 'age': 21, 'branch': 'CSE'}
```

Deleting from Dictionary in Python

- Use del keyword to remove specific key.
- Use .pop(key) to remove and return value.
- Use .clear() to remove all items.

Example

```
d = {"a": 1, "b": 2, "c": 3}
del d["b"]
print(d)
d.pop("a")
print(d)
d.clear()
print(d)
```

Output

```
{'a': 1, 'c': 3}
{'c': 3}
{}
```

Iterating a Dictionary in Python

- Dictionaries can be iterated with a for loop.

Example

```
car = {"brand": "Toyota", "model": "Innova", "year": 2020}
for key in car:
    print(key, ":", car[key])
```

Output

```
brand : Toyota
model : Innova
year : 2020
```

Dictionary Operators

List Operators in Python

in	It is known as membership operator. It returns True if a particular key is present in the specified dictionary.
not in	It is also a membership operator and It returns true if a particular dictionary key is not present in the list.

Example

```
d = {"x": 10, "y": 20}
print("x" in d)     # True
```

Dictionary Functions & Methods

Dictionary Functions & Methods in Python

- Python provides various in-built functions and methods which can be used with dictionary. Those are
  - len()
  - Keys()
  - values()
  - items()
  - update()

☞ len():

- In Python, **len()** function is used to find the length of dictionary ,i.e it returns the number of items in the dictionary.

Syntax: `len(dictionary)`

Example: dictlendemo.py

```
n={“x”:20,“y”:30}
print("length of dict :",len(n))
```

Output:

```
length of dict : 2
```

Dictionary Functions & Methods in Python Cont..

☞ keys ():

- In Python, keys() method returns all keys in the dictionary.

Syntax: `dictionary.keys()`

```
dict1={"name": "John", "age": 30}
print("Keys of dict1:",dict1.keys())
```

Output:

keys of dict1 : dict_keys(['name', 'age'])

Dictionary Functions & Methods in Python Cont..

☞ values ():

- In Python, values() method returns all values in the dictionary.

Syntax: `dictionary.values()`

```
dict1={"name": "John", "age": 30}
print("Values of dict1:",dict.values())
```

Output:

values of dict1 : dict_values(['John', 30])

Dictionary Functions & Methods in Python Cont..

☞ items ():

- In Python, items() method returns all keys-values in the dictionary.

Syntax: `dictionary.items()`

```
dict1={"name": "John", "age": 30}
print("Items of dict1:",dict1.items())
```

Output:

Items of dict1 : dict_items([('name', 'John'), ('age', 30)])

Dictionary Functions & Methods in Python Cont..

☞ update ():

- In Python, update() method merges dictionaries.

Syntax: `dictionary.update(dict)`

```
dict1={"name": "John", "age": 30}
dict2={"City": "Delhi"}
dict1.update(dict2)
print(dict1)
```

Output:

Items of dict1 : {'name': 'John', 'age': 30, 'city': 'Delhi'}