Understanding Dictionaries in Python

sainsdata.id

What is a dictionary in python?

A dictionary is a built-in data type that represents a collection of key-value pairs. It is also known as a hash map or associative array in other programming languages. Dictionaries are unordered, mutable (can be modified), and indexed by keys. The values within dictionaries can be of any data type, including lists and other dictionaries.

```
{
    key_1 :  value_1,
    key_2 :  value_2,
    key_3 :  [value_3_1, ...],
    key_4 :  {
                key_4_1 : value_4_1,
                ...
             },
    ...
}
```

How to create dictionary?

```
pikachu = {
    "name": "Pikachu",
    "type": "Electric",
    "hp": 35
}

charmander = dict(
    name="Charmander",
    type="Fire",
    hp=39
)

bulba_item = [
    ("name", "Bulbasaur"),
    ("type", "Grass/Poisson"),
    ("hp", 45)
]
bulbasaur = dict(bulba_item)
```

Using curly braces

using constructor **dict(...)**

convert from **list/set**

```
{'name': 'Pikachu', 'type': 'Electric', 'hp': 35}
{'name': 'Charmander', 'type': 'Fire', 'hp': 39}
{'name': 'Bulbasaur', 'type': 'Grass/Poisson', 'hp': 45}
```

More complex dictionary

```
charizard = {
    "name": "Charizard",
    "type": ["Fire", "Flying"],
    "height": 1.7,
    "weight": 90.5,
    "stats": {
        "HP": 78,
        "attack": 84,
        "defense": 78,
        "speed": 100,
        "total": 364,
        "special": {"sp.attack": 109,
                    "sp.defense": 85}
    }
}
```

list as value

another dictionary as value

Accessing the value

```
# accessing "name"
name = charizard["name"]

# accessing value of "type" (a list)
types = charizard["type"]

# accessing 1st element of "type"
type_1 = charizard["type"][0]

# accessing value of "stats" (a dictionary)
stats = charizard["stats"]

# accessing value of "attack"
attack = charizard["stats"]["attack"]

# accessing value of "special defence"
sp_def = charizard["stats"]["special"]["sp.defense"]


# accessing non-existent keys using []
# this will throw an error (KeyError)
species = charizard["species"]

# accessing non-existent keys using get(...)
# this will return a None
species = charizard.get("species")
```

There are two ways to access values in a dictionary, using square brackets **[...]** and the **get(...)** method.

The distinction is how they handle accessing non-existent keys.

[...] will throw an error
get(...) will return None

throw an error (**KeyError**)

return **None**

Modifying the value

```python
# change value of "name"
charizard["name"] = "Mega Charizard X"
# add an item into the list
charizard["type"].append("Dragon")
# change value of "attack"
charizard["stats"]["attack"] = 130
# change value of "sp.attack"
charizard["stats"]["special"]["sp.attack"] = 130
# add new key-value
charizard["species"] = "Flame Pokemon"
```

```
{
    'name': 'Mega Charizard X',
    'type': ['Fire', 'Flying', 'Dragon'],
    'height': 1.7,
    'weight': 90.5,
    'stats': {
        'HP': 78, 'attack': 130, 'defense': 78,
        'speed': 100, 'total': 364,
        'special': {'sp.attack': 130, 'sp.defense': 85}
    },
    'species': 'Flame Pokemon'
}
```

removing a key-value

```python
# removing key 'stats'
del charizard["stats"]
print("Charizard without stats :\n")
print(charizard)
```

```
{
    'name': 'Mega Charizard X',
    'type': ['Fire', 'Flying', 'Dragon'],
    'height': 1.7,
    'weight': 90.5,
    'species': 'Flame Pokemon'
}
```

removing item using keyword '**del**'

```python
# using pop method
# returning the popped item
popped_item = charizard.pop("type")
print(f"Removed item: {popped_item}\n")
print(charizard)
```

removing item using method '**pop**'

```
Removed item: ['Fire', 'Flying', 'Dragon']

{
    'name': 'Mega Charizard X',
    'height': 1.7,
    'weight': 90.5,
    'species': 'Flame Pokemon'
}
```

Combining two dictionaries

```python
pikachu = {
    "name": "Pikachu", "type": "Electric", "hp": 35
}

pikachu_stats = {
    "attack": 55,  "defense": 40, "speed": 90
}

# combine pikachu_stats into pikachu
pikachu.update(pikachu_stats)

print(f"Pikachu (combined) :\n{pikachu}")
```

```
Pikachu (combined) :

{
    'name': 'Pikachu',
    'type': 'Electric',
    'hp': 35,
    'attack': 55,
    'defense': 40,
    'speed': 90
}
```

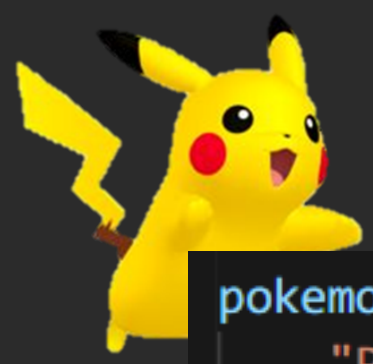
if there is a **duplicate key**, its value will be updated using the new one

(**hp = 0**) ====> (**hp = 45**)

```python
bulbasaur = {
    "name": "Bulbasaur", "type": "Poisson", "hp": 0
}

bulbasaur_stats = {
    "hp": 45, "attack": 49, "defense": 49, "speed": 45
}

# combine bulbasaur_stats into bulbasaur
bulbasaur.update(bulbasaur_stats)

print(f"Bulbasaur (combined) :\n{bulbasaur}")
```

```
Bulbasaur (combined) :

{
    'name': 'Bulbasaur',
    'type': 'Poisson',
    'hp': 45,
    'attack': 49,
    'defense': 49,
    'speed': 45
}
```

Iterating over a dictionary



```python
pokemon = {
    "Pikachu": "Electric",
    "Charmander": "Fire",
    "Bulbasaur": "Grass",
    "Squirtle": "Water",
}

# iteration using key
for key in pokemon:
    print(f"{key} : {pokemon[key]}")

print("======================")

# iteration using key-value
for key, value in pokemon.items():
    print(f"{key} : {value}")
```

```
Pikachu : Electric
Charmander : Fire
Bulbasaur : Grass
Squirtle : Water
======================
Pikachu : Electric
Charmander : Fire
Bulbasaur : Grass
Squirtle : Water
```