Dictionary

- Dictionaries are used to store values in key-value pairs.
- A dictionary is a collection which is **ordered**, **changeable** and **do not allow duplicates**.
- dictionaries are written with curly brackets and have keys and values.

```
dictionary = {"name":"vaibhav",
             "location":"mumbai",
               "yop":2022}
```

Dictionary Items

- Dictionary items are ordered, changeable and does not allow duplicates.
- Dictioanry items are presented in key-value pairs and can be reffered by using key name.

Ordered or Unordered

- When we say that dictionaries are ordered, it means that the items have a defined order, and that order will not change.
- Unordered means that the items does not have a defined order, you cannot refer to an item by using an index.

Changeable

- Dictionaries are changeable, means that we can change, add or remove items after the dictionary has been created.

Duplicates Not Allowed

- Dictionaries can not have more than one item with the same key.

Dictionary Length

- len() function is used to print the length of the dictionary.

```
len(dictionary)
```

```
4
```

Dictionary Items - data types

- The values in dictionary items can be of any data type.

```
thisdict = {
  "brand": "Ford",
  "electric": False,
  "year": 1964,
  "colors": ["red", "white", "blue"]
}
print(thisdict)
```

```
{'brand': 'Ford', 'electric': False, 'year': 1964, 'colors': ['red', 'white', 'blue']}
```

Dictionary Methods

copy()

- It will make copy of a particular directory.

syntax

new_dictionary = dictionary.copy()

```
1 copied_dict = dictionary.copy()
2 copied_dict
```

```
{'name': 'vaibhav', 'location': 'mumbai', 'yop': 2022}
```

fromkeys()

- This method creates a dictionary with specified keys and values.

syntax

- dictionary.fromkeys(key,value)

```
1 dictionary2 = {}
2 key = ('A','B','C')
3 value = 10
4 dictionary2.fromkeys(key,value)
```

```
{'A': 10, 'B': 10, 'C': 10}
```

get()

- This method will return values of specified key from a dictionary.
- You need to pass at least one 'key' as an argument else it will create error.

syntax

- dictionary.get('key')

```
1 dictionary.get("location")
```

```
'mumbai'
```

items()

- This command will return a list containing tuple with each key-value pair.

syntax

- dictionary.items()

```
1 dictionary.items()
```

```
dict_items([('name', 'vaibhav'), ('location', 'mumbai'), ('yop', 2022)])
```

keys()

- It will return a list of keys from the specified dictionary.

syntax

- dictionary.keys()

```
1 dictionary.keys()
```

```
dict_keys(['name', 'location', 'yop'])
```

values()

- It will return a list of values from the specified dictionary.

syntax

- dictionary.values()

```
dictionary.values()
```

```
dict_values(['vaibhav', 'mumbai', 2022])
```

update()

- This command is used to update the dictionaries with specified key-value pair.

syntax

- dictionary.update({'new_key':'value'})

```
dictionary.update({'year':2001})
```

```
dictionary
```

```
{'name': 'vaibhav', 'location': 'mumbai', 'yop': 2022, 'year': 2001}
```

pop()

- It will remove the element(key-value) with the specified key.
- You need to paas **at least one key** as an argument.

syntax

- dictionary.pop('key')

```
copied_dict.pop('yop')
```

```
2022
```

```
copied_dict
```

```
{'name': 'vaibhav', 'location': 'mumbai'}
```

popitem()

- It will removes the last inserted key-value pair from the specified dictionary.

syntax

- dictionary.popitem()

```
copied_dict.popitem()
```

```
('location', 'mumbai')
```

```
copied_dict
```

```
{'name': 'vaibhav'}
```

setdefault()

- It will return the value of the specified key.
- If the key is not exist, insert the key with the specified value.

syntax

- dictionary.setdefault('key')
- dictionary.setdefault('key',value')

```
1 copied_dict.setdefault('name')
```

```
'vaibhav'
```

```
1 copied_dict.setdefault("course","PG-DBDA")
```

```
'PG-DBDA'
```

```
1 copied_dict
```

```
{'name': 'vaibhav', 'course': 'PG-DBDA'}
```

clear()

- It will remove all the elements from the dictionary, including keys and values.

syntax

- dictionary.clear()

```
1 copied_dict.clear()
```

```
1 copied_dict
```

```
{}
```