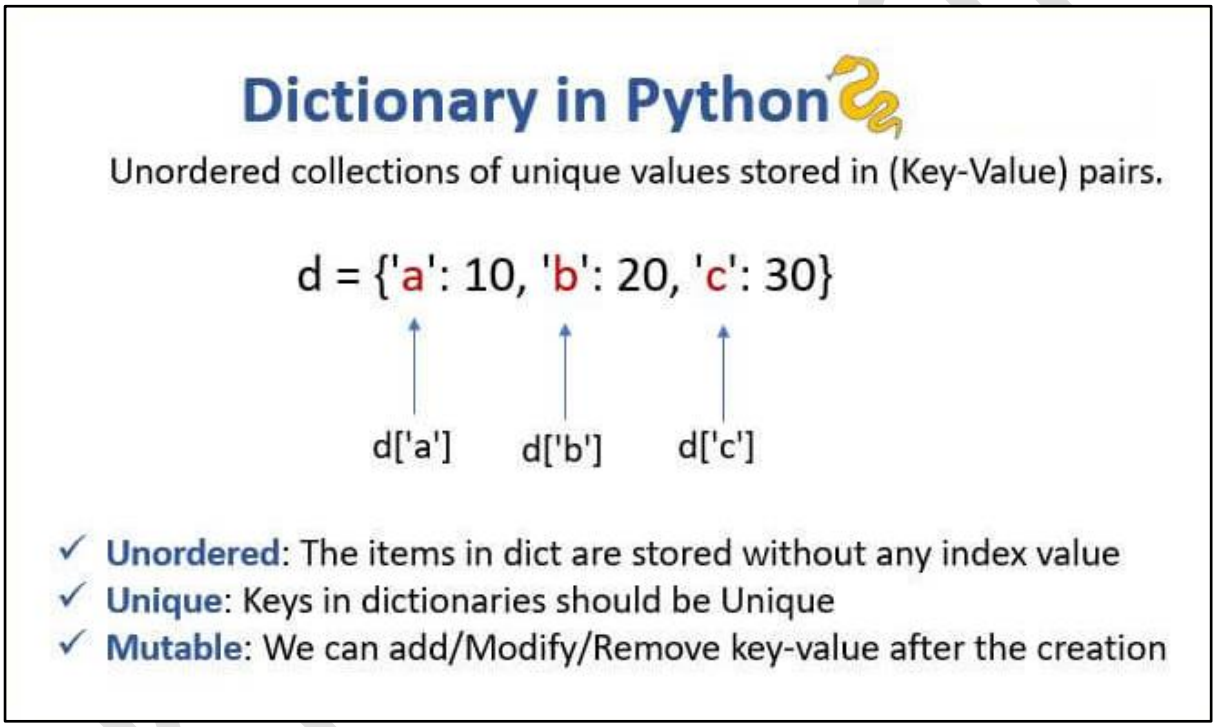
Table of Contents

Khan Omar

Chapter 5

Python Collections (Arrays)
There are four collection data types in the Python programming language:
-List is a collection which is ordered and changeable. Allows duplicate members.
-Tuple is a collection which is ordered and unchangeable. Allows duplicate members.
-Set is a collection which is unordered, unchangeable*, and unindexed. No duplicate members.
-Dictionary is a collection which is ordered** and changeable. No duplicate members.

Dictionaries




```
# Dictionary items are ordered, changeable, and does not allow duplicates.
# Dictionary items are presented in key:value pairs, and can be referred to by using the key name.
car1 = {
  "brand": "Ford",
  "model": "Mustang",
  "colors": ["red", "white", "blue"],
  "electric": False,
  "year": 1963
}
# {'brand': 'Ford', 'model': 'Mustang', 'colors': ['red', 'white', 'blue'], 'electric': False, 'year': 1963}
print(car1)
```

```
# car1["year"]=1964
# Or
car1.update({"year": 1964})
# {'brand': 'Ford', 'model': 'Mustang', 'colors': ['red', 'white', 'blue'], 'electric': False, 'year': 1964}
print(car1)
```

Dictionary Creation

```
mySelf = {
   "name": "Omar",
   "title": "Oily",
   "year": 2004,
   "Madistic": {
      "name": "Madi",
      "title": "Perfect",
      "year": 2005,
   }
}
```

Access Items

```
print(mySelf["name"])  # Omar
print(mySelf["Madistic"]["name"])  # Madi
```

Dictionaries methods

```
print(mySelf.get("year"))  # 2004
print(mySelf.keys())  # dict_keys(['name', 'title', 'year', 'Madistic'])
# dict_values(['Omar', 'Oily', 2004, {'name': 'Madi', 'title': 'Perfect', 'year': 2005}])
print(mySelf.values())
# dict_values(['Omar', 'Oily', 2004, {'name': 'Madi', 'title': 'Perfect', 'year': 2005}])
# The items() method will return each item in a dictionary, as tuples in a list.
# dict_items([('name', 'Omar'), ('title', 'Oily'), ('year', 2004), ('Madistic', {'name': 'Madi', 'title': 'Perfect', 'year': 2005})])
print(mySelf.items())
# The update() method will update the dictionary with the items from the given argument.
# The argument must be a dictionary, or an iterable object with key:value pairs.
updatemySelf = {
   "hobby": "Football",
   "topper": True
}
```

Khan Omar

```
# mySelf["Madistic"].update(updatemySelf) # {'name': 'Omar', 'title': 'Oily', 'year': 2004, 'Madistic': {'name': 'Madi', 'title': 'Perfect', 'year': 2005, 'hobby': 'Football'}}

mySelf.update(updatemySelf)
print(mySelf)  # {'name': 'Omar', 'title': 'Oily', 'year': 2004, 'Madistic': {'name': 'Madi', 'title': 'Perfect', 'year': 2005}, 'hobby': 'Football' 'topper':True}
```

Remove Items

```
# The pop() method removes the item with the specified key name.
mySelf.pop("topper")
print(mySelf)

# The popitem() method removes the last inserted item (in versions before 3.7, a random item is removed instead)
updatemySelf = {
   "topper": True
}
mySelf.update(updatemySelf)
del mySelf['topper']
# {'name': 'Omar', 'title': 'Oily', 'year': 2004, 'Madistic': {'name': 'Madi', 'title': 'Perfect', 'year': 2005}, 'hobby': 'Football'}
print(mySelf)

# The del keyword can also delete the dictionary completely:
car2 = {
   "brand": "Ford",
   "model": "Mustang",
   "year": 1964
}
print(car2)
del car2
print(car2) # this will cause an error because "thisdict" no longer exists.

# car3 = dict(car1)
# Or
car3 = car1.copy()
print(car3) # {'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

Khan Omar