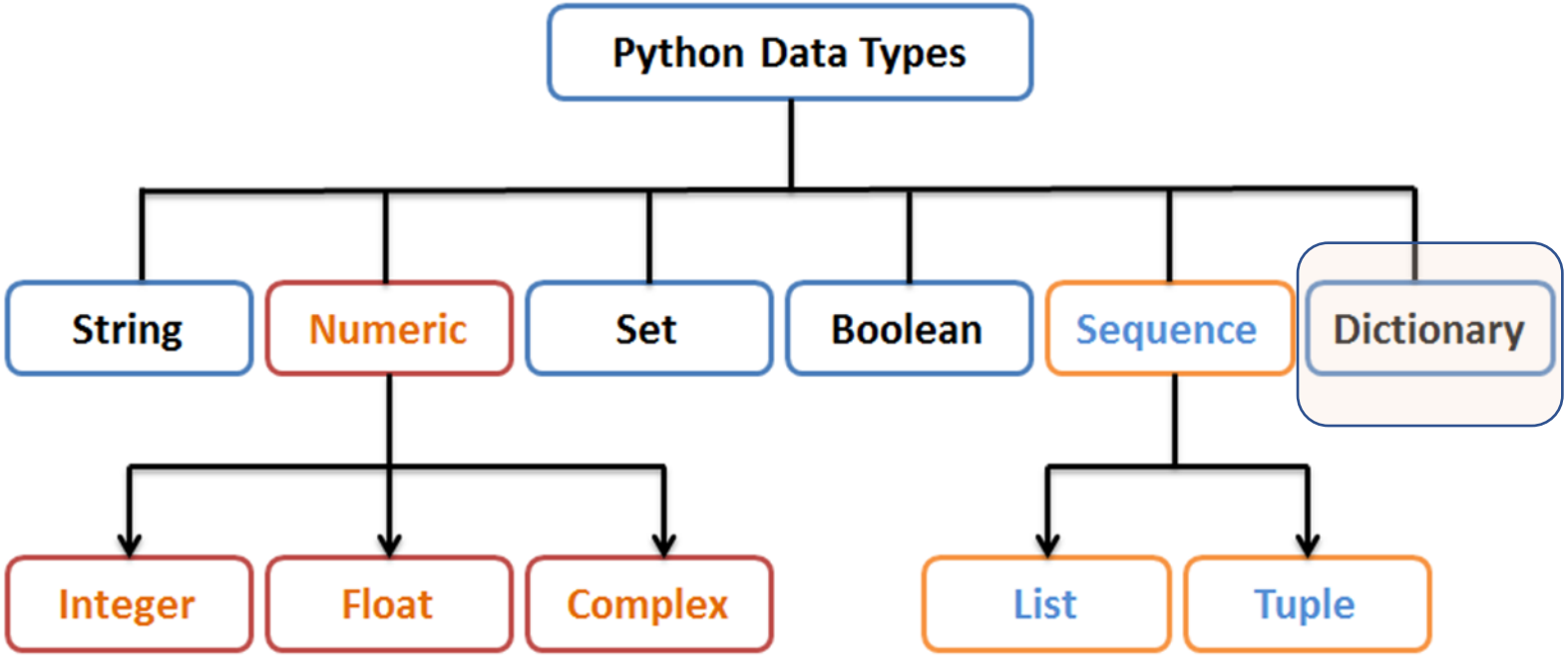
Python Dictionary

Anoop S Babu
Faculty Associate
Dept. of Computer Science & Engineering
bsanoop@am.amrita.edu

Dictionary


Python Data Types
String
Numeric
Set
Boolean
Sequence
Dictionary
Integer
Float
Complex
List
Tuple

Dictionary

- Compound data type that allows to work with key value pair
- Each item is a Key-Value pair
- Keys are unique
- Keys are immutable.
  - Strings, numbers or tuples allowed. No list.

Creating empty Dictionary

- Using curly braces {}

```
>>> myDict = {}
>>> type(myDict)
<class 'dict'>
```

- Using keyword dict()

```
>>> myDict = dict()
>>> myDict
{}
```

Creating dictionary with elements

```
>>> myDict = {"Program":"MCA", "Course":"Python", "Semester":1}
>>> print(myDict)
{'Program': 'MCA', 'Course': 'Python', 'Semester': 1}
# integers as keys
>>> courses = {1:"Python", 2:"Java", 3:"OS"}
>>> print(courses)
{1: 'Python', 2: 'Java', 3: 'OS'}
```

Dictionary comprehension

- Can create using dictionary comprehension

```
>>> squares = {x:x**2 for x in range(1,6)}
>>> print(squares)
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

Unique keys

```
>>> courses = {1:"Python", 2:"Java", 3:"OS", 2:'DBMS'}
>>> print(courses)
{1: 'Python', 2: 'DBMS', 3: 'OS’}
```

- **No duplicate key** is allowed. If any, **last value** is assigned

Accessing values

- Using keys as indices

```
>>> myDict = {"Program":"MCA", "Course":"Python", "Semester":1}
>>> print(myDict["Course"])
Python
```

- Using get() method

```
>>> print(myDict.get("Program"))
MCA
```

Changing values

- Use keys as indices

```
>>> courses = {1:"Python", 2:"Java", 3:"OS"}
>>> courses[3] = "DBMS"
>>> print(courses)
{1: 'Python', 2: 'Java', 3: 'DBMS'}
```

Traversing a dictionary - 1

- Using items() method

Syntax: for key, val in *dict*.items():

print(key, ":", val)

```
>>> courses = {1:"Python", 2:"Java", 3:"OS"}
>>> for key,value in courses.items():
        print("{}:{}".format(key,value))
1:Python
2:Java
3:OS
```

Traversing a dictionary - 2

- Using keys() method

Syntax: for key in *dict*.keys():

print(*dict*[key])

```
>>> courses = {1:"Python", 2:"Java", 3:"OS"}
>>> for key in courses.keys():
        print(courses[key])
Python
Java
OS
```

Traversing a dictionary - 3

- Using values() method

Syntax: for val in *dict*.values():

print(val)

```
>>> courses = {1:"Python", 2:"Java", 3:"OS"}
>>> for value in courses.values():
        print(value)
Python
Java
OS
```