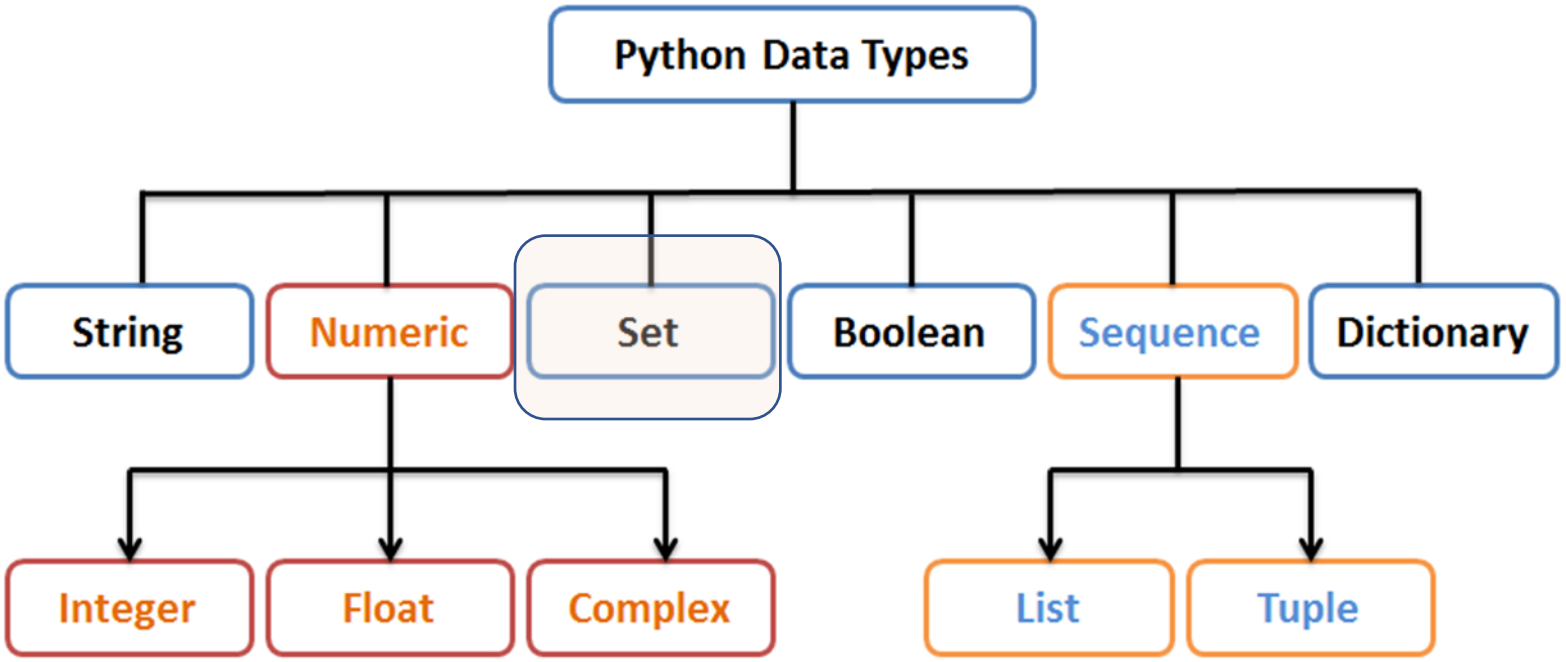
Python Set

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

Sets


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

Sets

- Unordered collection of unique hashable elements
- Do not support indexing and slicing
- Mutable
- Allow multiple data types

3 things to remember about set

- Items of the set are **not in any order**.
- **No duplicate** items are allowed in set.
- **Items** in the set **must be immutable** objects.

Creating Sets

- Using curly braces {}

```
>>> numbers = {1,3,8,2,1,9}      #duplicate values
>>> numbers
{1, 2, 3, 8, 9}                  #unique values
```

Creating Sets

- Using set() function

```
>>> vowels = set("aeiou")
>>> vowels
{'e', 'i', 'o', 'u', 'a'}
>>> numbers = set([2,4,1,2,3,1])
>>> numbers
{1, 2, 3, 4}
```

Creating Sets

- Using set comprehension

```
>>> even = {x for x in range(1,11) if x%2 == 0}
>>> even
{2, 4, 6, 8, 10}
```

Creating empty sets

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

```
>>> s = set()
>>> type(s)
<class 'set’>
```

- Create empty sets with set() function without arguments

Sets – Immutable elements

```
>>> numbers ={2,4,[1,3]}
…
TypeError: unhashable type: 'list'
```

- Set cannot contain mutable object as element
- No lists or dictionaries as set elements
- Allows tuple

```
>>> numbers = {1,(2,3),5}
>>> numbers
{(2, 3), 1, 5}
```

Sets – no indexing

```
>>> numbers = {1,2,3,4}
>>> numbers[1]
…
TypeError: 'set' object is not subscriptable
```

- No indexing as it’s unordered

Adding elements to sets

- add() method adds a single element

```
>>> numbers = set()
>>> numbers.add(2)
>>> numbers.add(3)
>>> numbers
{2, 3}
```

- update() method adds multiple elements

```
>>> numbers.update((6,4))
>>> numbers
{2, 3, 4, 6}
```

Removing elements from a set

- remove(obj) - removes the object

```
>>> numbers = {2,6,1,3}
>>> numbers.remove(6)
>>> numbers
{1, 2, 3}
>>> numbers.remove(4)
…
KeyError: 4
```

- Raise an error if element is not in the set

Removing elements from a set

- discard(obj) - removes the object

```
>>> numbers = {2,6,1,3}
>>> numbers.discard(1)
>>> numbers
{2, 3, 6}
>>> numbers.discard(4)
>>>
```

- Do not raise an error if element is not in the set

Removing elements from a set

- pop() - removes a random item from the set
- Takes no argument
- Returns the deleted item

```
>>> numbers = {2,6,1,3}
>>> numbers.pop()
1
```

Removing elements from a set

- clear() - clear the set

```
>>> numbers = {2,6,1,3}
>>> numbers.clear()
>>> numbers
set()
```

Traversing a set

```
>>> numbers = {2,6,1,3}
>>> for n in numbers:
        print(n)
```

```
1
2
3
6
```