Introduction to PYTHON

Module 1 / Lecture-3

By: Atul Kumar Uttam

Assistant Professor

Computer Engineering & Applications Department,

GLA University, Mathura

Topics

- Data types in Python
- Numbers
  - Integer
  - Float
  - Complex
- Sequence
  - List
  - Tuple
  - String
- Set
- Dictionary
- Boolean

Data Type

- Every value in Python has a data-type.
- Everything is an object in Python programming

Data types	**Classes**
Variables	**Instance (object)**

Data type

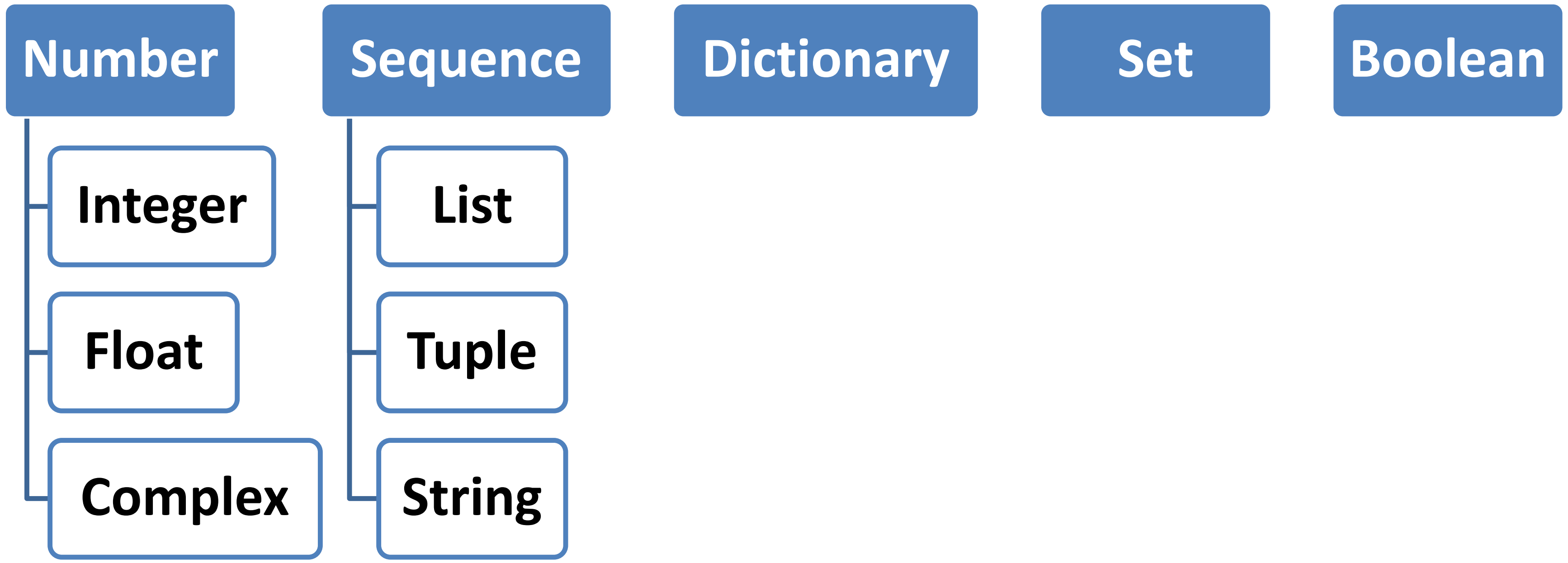
- The data type is an attribute of data, that tells a programming language interpreter/compiler how the programmer mean to use it.

- Values are classified into different **data types (Classes)**
- **type()** function returns the data type (class) of a variable or a value.
- **isinstance()** function returns boolean value (True/False), check if an object belongs to a particular class.

```
>>> type("Hello, World!")
str

>>> isinstance(10, int)
True
```

Classification of Data Type in Python



Numbers

Class: **int, float complex**

Ex1.
```
a = 5
print(a, "is of type", type(a))  # int
```

Ex2.
```
b = 2.0
print(b, "is of type", type(b))  #float
```

Ex3.
```
c = 1+2j
print(isinstance(c,complex))  #True
```

- Integers can be of any length, it is only limited by the memory available.
- A floating point number is accurate up to 15 decimal places

Sequence

- Class: **list** **tuple** **string**
- Ordered collection of elements
- Elements can be same or different data types
- **List**

```
>>>a = [1, 2, ‘z’]
```

- **Tuple**

```
>>>b = (1, 2, ’z’)
```

- **String**

```
>>>c = ‘A’
>>>d = “AAA”
>>>e = “””This is also a string”””
```

List [ ]

- An ordered sequence of elements.
- Elements of an list can be of same / different types.
- Elements are separated by commas
- Enclosed within brackets [ ]

```
>>> a = [1, 2.2, 'python']
```

- We can use the **slicing operator [ ]** to extract an item or a range of items from a list.
- Index starts form 0 in Python(forward index left to right).
- Index can be –ve (backward index right to left)

```
>>>a=[10,20,30,40,45,65,66]

>>>print(a[2])
30

>>>print(a[0:3])
[10, 20, 30]

>>>print(a[2:4])
[30, 40]
```

```
>>>print(a[2:])
[30, 40, 45, 65, 66]

>>>print(a[1:-3])
[20, 30, 40]

>>>print(a[-6:-3])
[20, 30, 40]
```

- Lists are ***mutable***,
- i.e. value of elements of a list can be altered.

```
>>> a = [10,20,30]

>>> a[2]=40

>>> a
[10, 20, 40]
```

Tuple ( )

- An **ordered sequence** of elements
- **Immutable**: i.e. tuples once created cannot be modified.
- It is defined within parentheses ()
- Elements are separated by commas.

```
>>> t = (5,'program', 1+3j)
```

- We can use the **slicing operator [] to extract items** but we **cannot change its value**.

```
t = (5,'program', 1+3j)

# t[1] = 'program‘
print("t[1] = ", t[1])

# t[0:3] = (5, 'program', (1+3j))
print("t[0:3] = ", t[0:3])

# Generates error
# Tuples are immutable
t[0] = 10
```

Strings

- String is sequence of **Unicode characters**.
- We can use single quotes or double quotes to represent strings.
- Multi-line strings can be denoted using triple quotes, ’’’ or ’’’.

>>> s = "This is a string"

>>> s = '''a multiline’’’

- Slicing operator [ ] can be used with string.
- Strings are **immutable**.

```
s = 'Hello world!‘

# s[4] = 'o‘
print("s[4] = ", s[4])

# s[6:11] = 'world‘
print("s[6:11] = ", s[6:11])

# Generates error
# Strings are immutable in Python
s[5] ='d'
```

Set

- **Unordered collection** of **unique elements**.
- Elements are separated by comma inside braces **{ }**.

```
>>>a = {5,2,3,1,4}

>>>print(a)
{5,2,3,1,4}

>>>print(type(a))
set
```

- Since, set are unordered collection, indexing has no meaning.
- Slicing operator [] does not work.
- Can perform set operations like union, intersection on two sets.
- Set have unique values.
- They eliminate duplicates.

```
>>>a ={11, 22}
>>> b={12,34,11}
>>>a.union(b)
   {34, 22, 11, 12}
>>> a.intersection(b)
{11}
>>> a.difference(b)
{22}
```

Dictionary

- Unordered
- One element is a: **Key-value** pair.
- Dictionaries are optimized **for retrieving data**.
- We **must know the key** to retrieve the value.
- Defined within braces **{}**

  {**key : value}**

- Key must be an immutable object
- Key can not be repeated
- Value can be of any type.
- Value can be repeated

```
d={10:'value', 'key':20}

>>>print(type(d))
<class 'dict'>

>>>print(d[10])
value

>>>print(d['key'])
20
```

Boolean

- Has two values: True/False
- Python returns boolean values:
  - While evaluating an expression
  - Comparing two values

```
>>>a=10
>>>b=10
>>>a==b
True
```

THANK YOU !!!