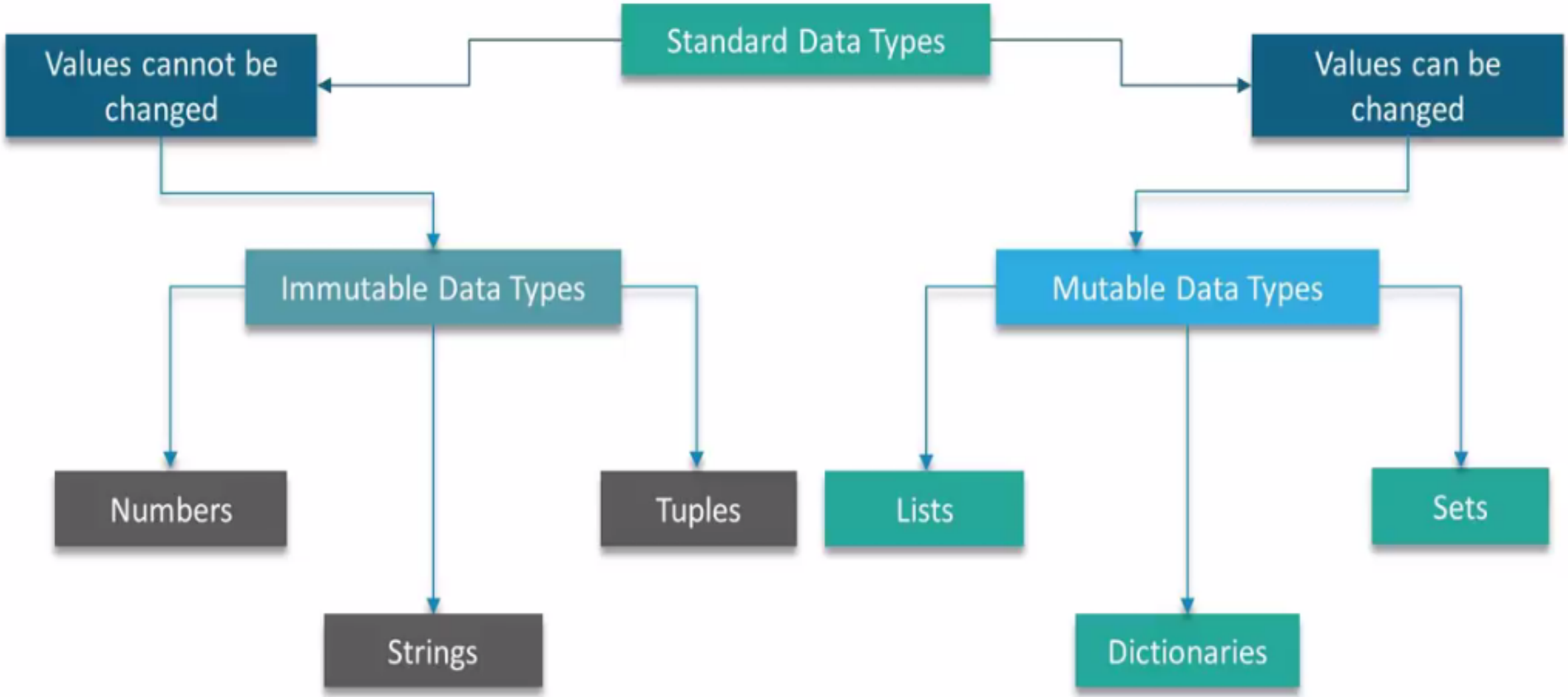
Data Types and Operators

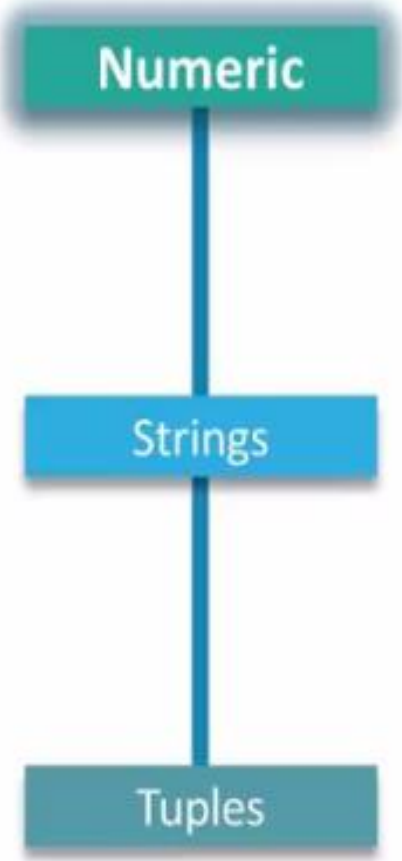
Dr. D. Sharma

Associate Professor, NMIMS Indore

Standard Data Types



Numeric Data Type – Immutable Data Type



Python supports three different numerical values

int (Signed Integers)

Complex Numbers

float (Real Numbers)

In Python you can represent **Numbers** in multiple ways:

- *Binary*
- *Octal*
- *Hexadecimal*

Strings – Immutable Data Type

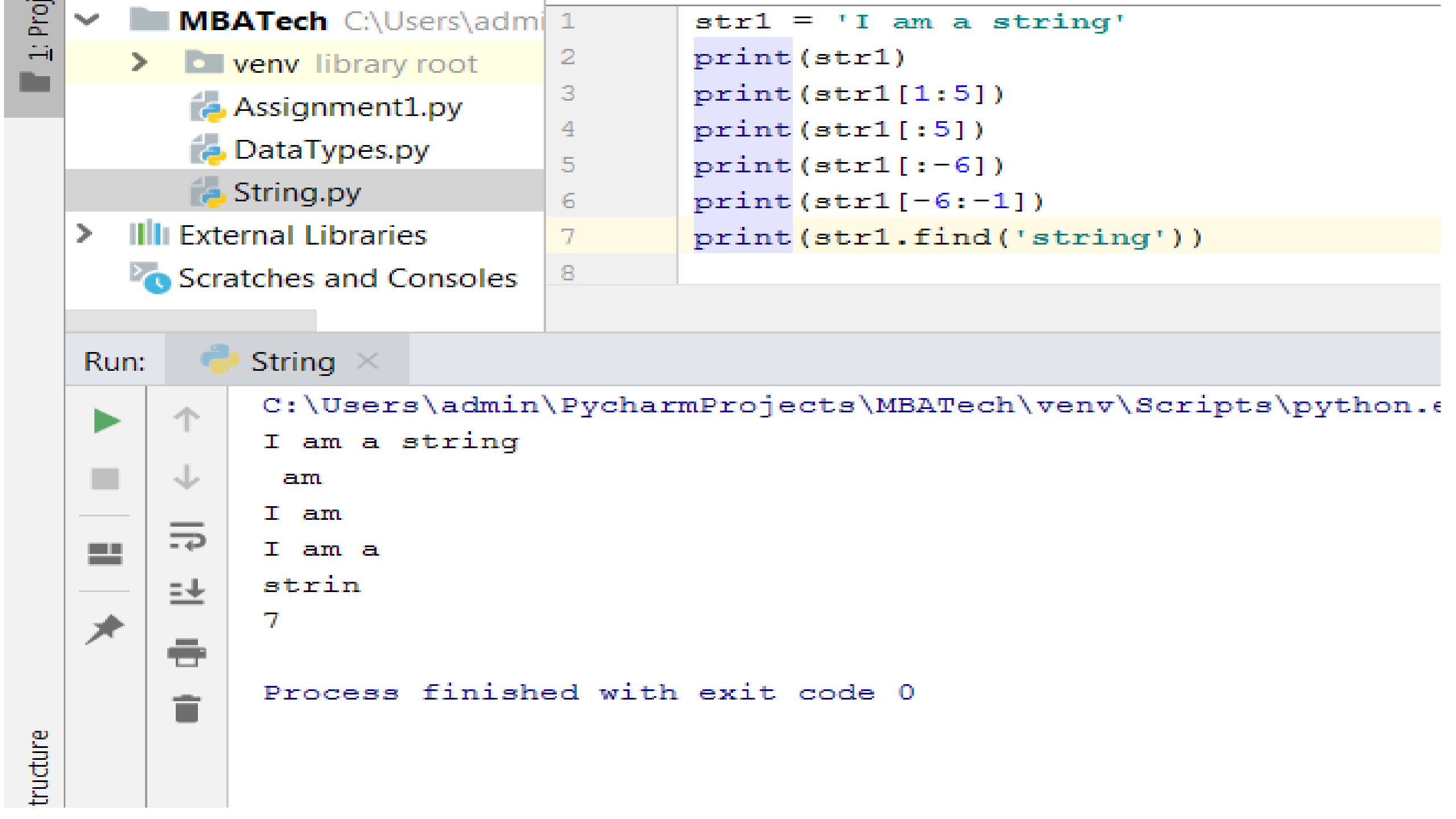
The continuous set of characters represented within quotation is called as String. Python allows for either pairs of single or double quotes. Python does not support a character type, these are treated as strings of length one

```
A='Welcome To edureka!'

B="Python is Great"

print(A)

print(B)
```

Python cannot differentiate between single and double quotes

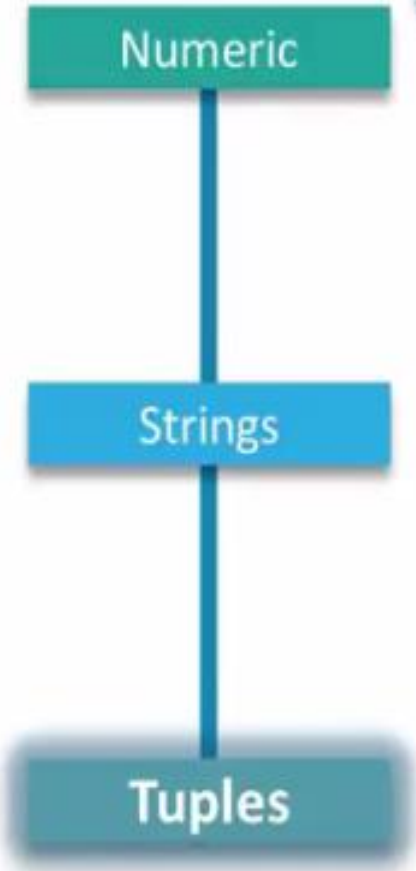
```
Welcome To edureka!
Python is Great
```

Output


MBATech C:\Users\admi
venv library root
Assignment1.py
DataTypes.py
String.py
External Libraries
Scratches and Consoles
1 str1 = 'I am a string'
2 print(str1)
3 print(str1[1:5])
4 print(str1[:5])
5 print(str1[:-6])
6 print(str1[-6:-1])
7 print(str1.find('string'))
8
Run:
String
C:\Users\admin\PycharmProjects\MBATech\venv\Scripts\python.e
I am a string
am
I am
I am a
strin
7
Process finished with exit code 0

Tuples – Immutable Data Type

Tuples consists of a number of values separated by comma. It is enclosed within parenthesis




```
A=(1,2,3.15,'edureka!')

print(A)
```

A Tuple can have objects of different data types, unlike Arrays in 'C'

```
(1, 2, 3.15, 'edureka!')
```

Output

Lists - Mutable Data Type

List is an ordered set of elements enclosed within square brackets. The main differences between Lists and Tuples are:

- Lists are enclosed in brackets[] and Tuples are enclosed within parenthesis()
- Lists are Mutable and Tuples are Immutable
- Tuples are faster than Lists

```
A=[1,2,3.15,'edureka!']

print(A)
```

Lists are enclosed within square brackets

```
[1, 2, 3.15, 'edureka!']
```

Output

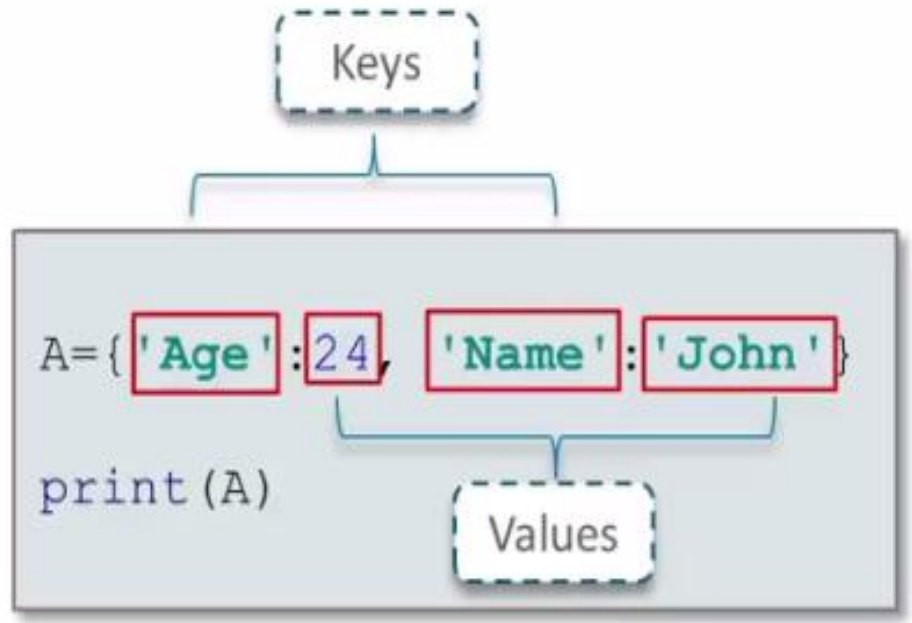
Dictionaries - Mutable Data Type

Lists

Dictionaries

Sets

Dictionaries contain key value pairs. Each key is separated from its value by a colon (:), the items are separated by comma, and the whole thing is enclosed within curly braces




```
{'Age': 24, 'Name': 'John'}
```

Output

Sets - Mutable Data Type

A set is an unordered collection of items. Every element is unique. A set is created by placing all the items (elements) inside curly braces {}, separated by comma.

Every element in a Set has to be unique

```
A={1,2,3,3}

print(A)
```

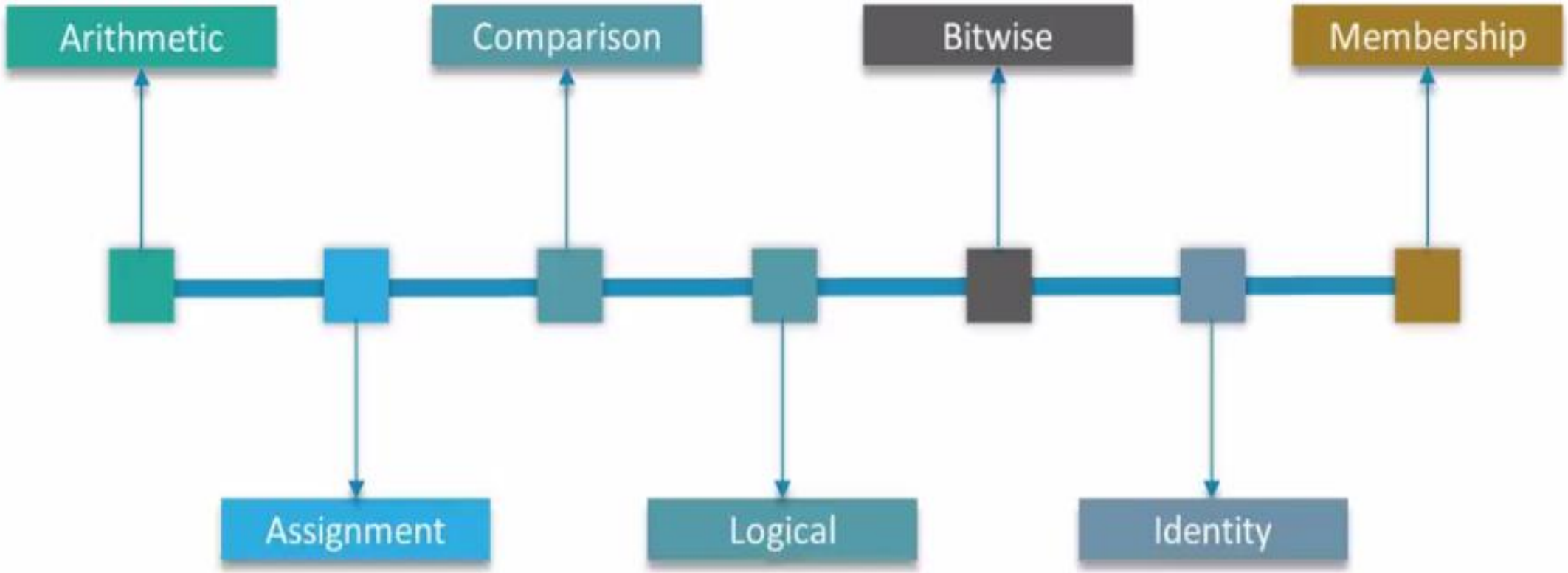
```
{1, 2, 3}
```

Output – Notice that 3 has appeared only once

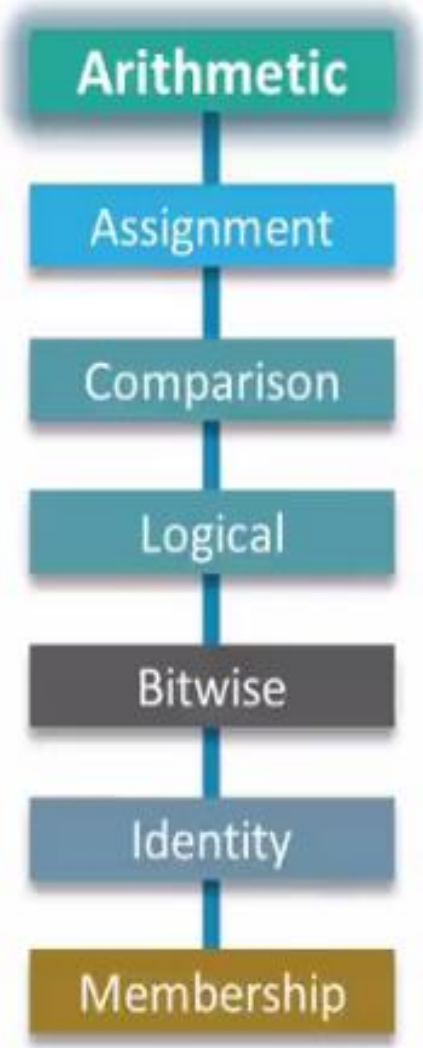
You can also create a Set by calling an in built-function **'set'**

Operators

Operators are the constructs which can manipulate the values of the Operands. Consider the expression 2 + 3 = 5, here 2 and 3 are Operands and + is called Operator

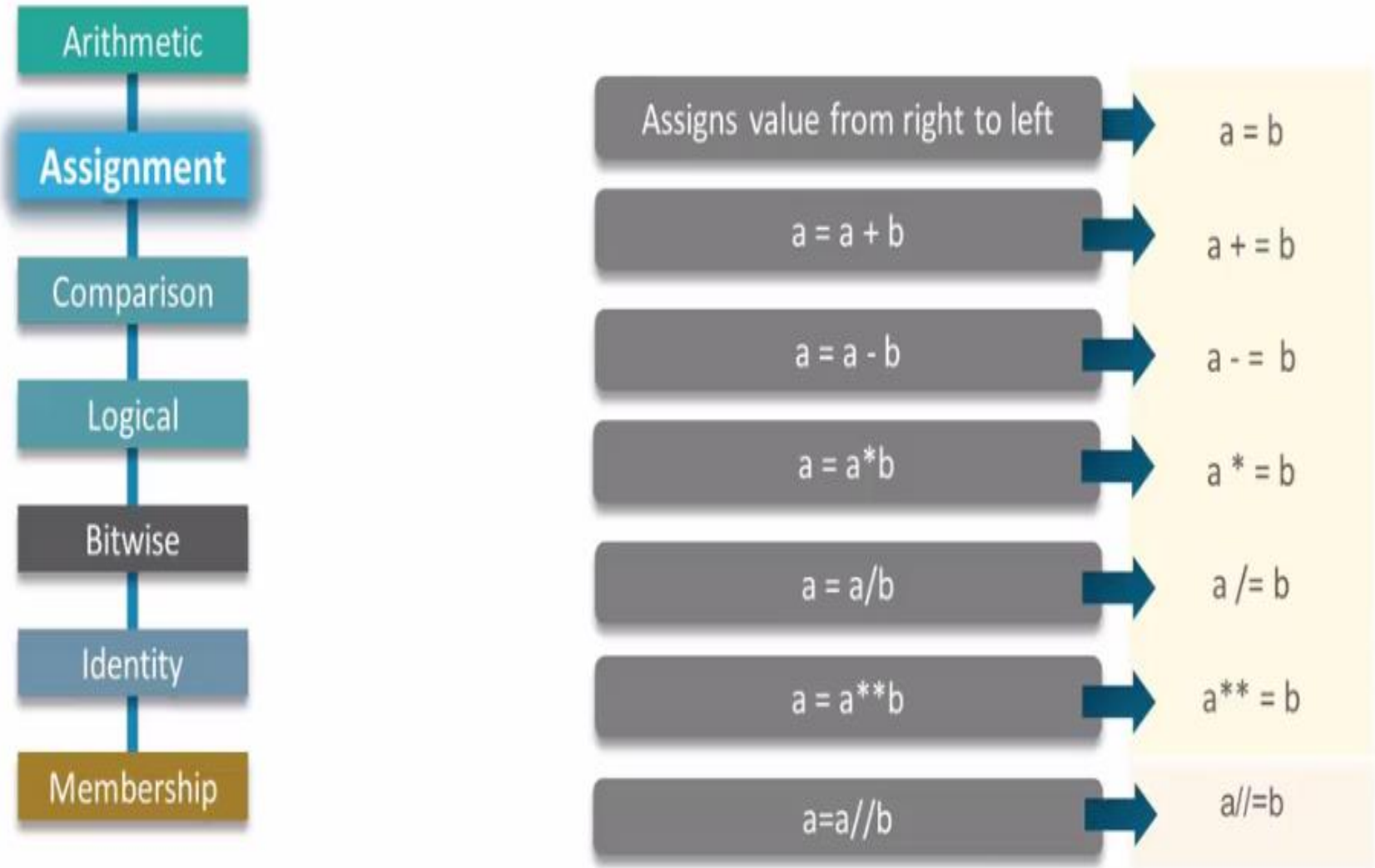


Arithmetic Operators

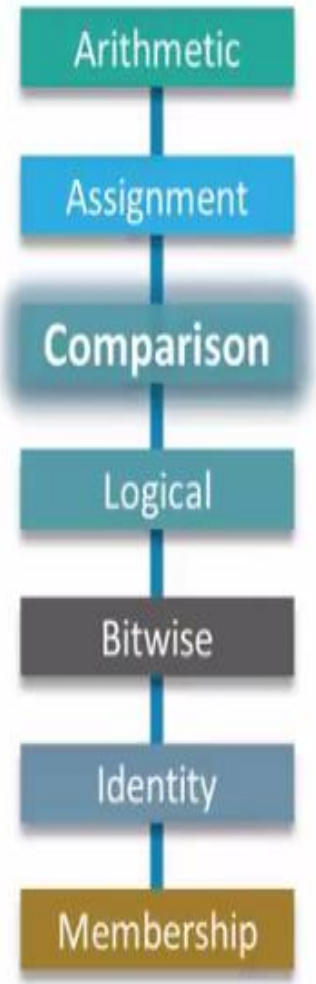


Addition	a + b
Subtraction	a – b
Multiplication	a * b
Division	a / b
Modulus	a % b
Exponent	a ** b
Floor Division	a // b

Assignment Operators



Comparison Operators

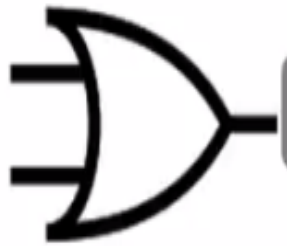


Equal To	a == b
Not Equal To	a != b
Greater Than	a > b
Less Than	a < b
Greater Than Equal To	a >= b
Less Than Equal To	a <= b

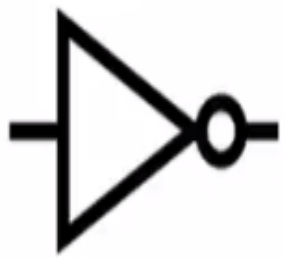
Logical Operators

- Arithmetic
- Assignment
- Comparison
- **Logical**
- Bitwise
- Identity
- Membership

a and b → Returns a if a is False, b otherwise

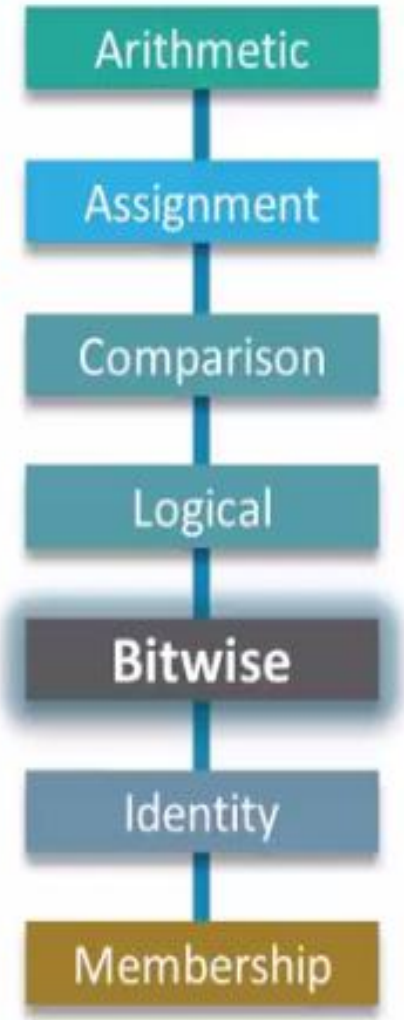


a or b → Returns b if b is False, a otherwise



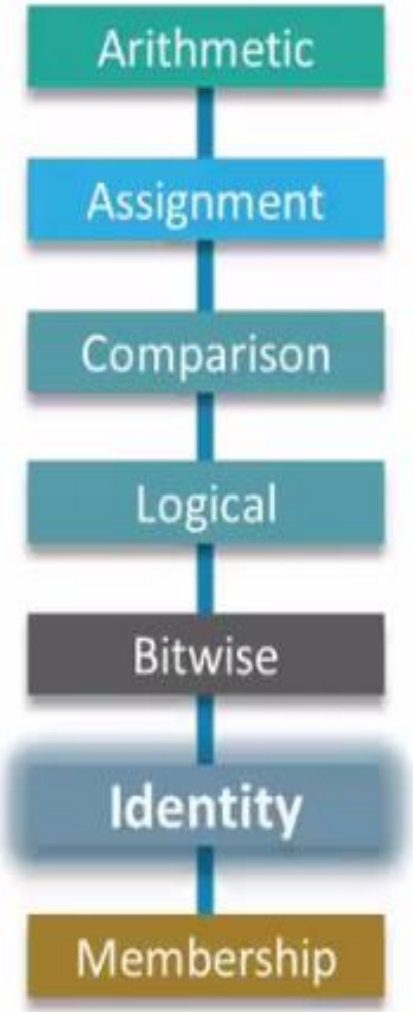
not a → Returns True if a is True, False otherwise

Bitwise Operators



Binary AND	a & b
Binary OR	a | b
Binary XOR	a ^ b
Binary NOT	a ~ b
Binary Left Shift	a <<
Binary Right Shift	a >> b

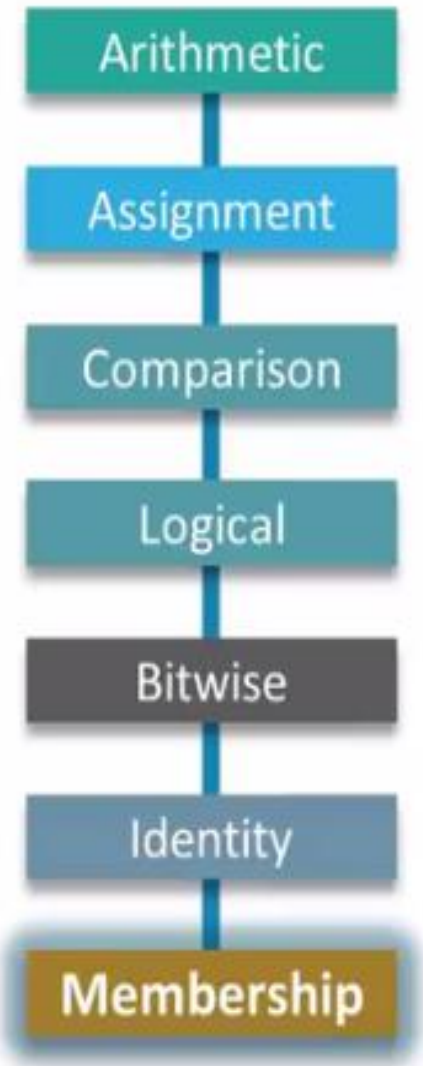
Identity Operators



is → Evaluates to TRUE if the variables on either side of the operator point to the same object and FALSE otherwise

is not → Evaluates to FALSE if the variables on either side of the operator point to the same object and TRUE otherwise

Membership Operators



in	Evaluates to TRUE if it finds a variable in the specified sequence and FALSE otherwise
not in	Evaluates to TRUE if it does not find a variable in the specified sequence and FALSE otherwise

- Thanks