Python Operators:

- Operator is a symbol which performs operations.
- In python we have different types of operators. They are like,

1. Arithematic Operators
2. Relational Operators
3. Logical Operators
4. Assignment Operators
5. Short-hand Operators
6. Identity Operators
7. Membership Operators

etc...

1. Arithematic Operators:

- + , - , * , / , % , // , ** operators are comes under arithematic operators.
- Arithematic operators are used to perform mathematical operations.
- It always gives values as a result.

For example,

```
>>> 10 + 2
12
>>> 10 - 2
8
>>> 10 * 2
20
>>> 10 / 2   # it returns result as float type
5.0
>>> 10 % 2
0
>>> 10 // 2    # it returns result as int type
5
>>> 10 ** 2
100
>>>
```

2. Relational Operators:

- < , <= , > , >= , == , != operators are called as Relational Operators.
- Relational operators are used to check conditions.
- Relational operators always gives us boolean (True | False) values as a result.

For example,

```
>>
>>> x = 10
>>> y = 20
>>> x < y
True
>>> x <= y
True
>>> x > y
False
>>> x >= y
False
>>> x == y
False
>>> x != y
True
```

3. Logical Operators:

- and , or, not keywords are called as logical operators.
- If you want to combine more than one condition result then we have to use logical operators.
- in our requirement, if all conditions has to satisfy then use 'and' operator.
- in our requirement, if any condition has to satisfy then use 'or' operator.

For example,

```
username == 'Srinivas' and password == 'Python'
s1 >= 35 and s2 >= 35 and s3 <= 35
>>> x = 20
>>> y = 10
>>> z = 40
>>> y < x and x < z
```

```
True
>>> y > x and  x > z
False
>>> y > x or  x < z
True
>>> y < x or  x < z
True
>>> y > x or  x > z
False
```

4. Assignment Operators:

- = symbol is called assignment operator.
- It is also called as right to left operator.
- Assignment operator is used to assign values to variables.

Syntax:

variable = value | variableName | Expression | function_call | method_call |

For example,

```
a = 10 # value
b = a  # variable
c = a + b # Expression
d = sum(4,5) # function_call
```

Example,

```
>>> def sum(x,y):
        return x + y
>>> d = sum(4,5) #  function_call
>>> d
9
```

5. Short-hand assignment Operators:

- += , -= , *= , /= , %= , //= , **= are called as Short-hand operators.

For example,

```
>>> x = x + 10   or   x += 10
>>> x = x - 10   or   x -= 10
>>> x = x * 10   or   x *= 10
```

6. Membership Operators:

- Python has membership operators, which test for membership in a sequence, such as strings, lists, or tuples.
- There are two membership operators explained below.

Operator		Description
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 finds a variable in the specified sequence and false otherwise.

For example,

```
>>> st = 'Python'
>>> 'P' in st
True
>>> 'z' in 'Python'
False
```

Example:

```
>>> lst = [1,2,3,4,'Python',True]
>>> 4 in lst              True
>>> 10 in lst                  False
>>> 3 not in lst          False
>>> 20 not in lst         True
```

Identity Operators:

- Identity operators compare the memory locations of two objects.
- There are two Identity operators explained below,

Operator		Description
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.

Example 1:

```
>>> st="Sai"
>>> id(st)                  57264032
>>> st1="Sai"
>>> id(st1)                 57264032
>>> st is st1                    True
>>> st is not st1                False
```

example 2,

```
>>> a = 10
>>> id(a)
1705600320
>>> b = 10
>>> id(b)
1705600320
>>> a is b     True
```

Example 3:

```
>>> lst=[1,2,3,4]
>>> id(lst)                 57297136
>>> lst1=[1,2,3,4]
>>> id(lst1)                     57240512
>>> lst is lst1                  False
>>> lst is not lst1              True
```