OPERATORS

Types Of Operators

- Arithmetic Operaotrs
- Relational or Comparison Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators
- Special Operators

--

Arithmetic Operators

- Addition +
- Subtraction -
- Multiplication *
- Division / (always generate float value)
- Integer Division or Floor Division // (always generate values in typecasted form)
- Modulo %
- Exponent or Power **

```
a = 10
b = 2

print(a+b)  #output : 12
print(a-b)  #output : 8
print(a*b)  #output : 20
print(a/b)  #output : 5.0
print(a%b)  #output : 0
print(a**b) #output : 1024
print(a//b) #output : 5
```

--

Relational or Comparison Operators

- Greater than >
- Smaller than <
- Greater than or equal to >=
- Smaller than or equal to <=
- Equal to ==

- Not equal to !=

```
a = 23
b = 56

print(a>b)  #output : F
print(a<b)  #output : T
print(a>=b) #output : F
print(a<=b) #output : T
print(a==b) #output : F
print(a!=b) #output : T

# Also applicable to string datatype and boolean datatype
# We can chain relational operators
```

--

Logical Operators

- and
- or
- not

For Boolean values

```
t = True
f = False

print(t and f) #output : False
print(t or f)  #output : True
print(not t)   #output : False
```

--

Bitwise Operator

- AND &
- OR |
- XOR ^
- Negation ~
- Right Shift >>
- Left Shift <<

Only applicable for int and booloean datatype

```
& ==> if both bits are 1 then only 1 otherwise 0
| ==> if atleast one bit is 1 then 1 otherwise 0
^ ==> if both the bits are different then 1 otherwise 0
~ ==> bitwise complement operator

>> ==>bitwise right shift
<< ==>bitwise left shift
```

--

Compound Assignment Operator

- += -= /= = //= *=
- &= |= ^= >>= <<=

--

Ternary Operator

(condition)?True Expression:False Expression

```
x = (10<20)?3:4
print(x) #output : 4
```

--

Special Operators

1.Identity Operator

- is #address comparison
- is not

```
a = 10
b = 10
c = 20

print(a is b) #output : True
print(a is c) #output : False
print(a is not b) #output : False
print(a is not c) #output : True

# Same behaviour for string
# Does not works for list
```

2.Membership Operator

- in #use to find whether an element belongs to a list or not
- not in

```
list1 = [10,2,5,'ad']

print(10 in list1) #output : True
print(2 not in list1) #output : False
print(50 in list1) #output : False
print(50 not in list1) #output : True

# Works for all kind of sequence datatype
```

--

Operator Precedence

```
()              Parentheses
**              Exponentiation
+x  -x  ~x      Unary plus, unary minus, and bitwise NOT
*  /  //  %     Multiplication, division, floor division, and modulus
+  -            Addition and subtraction
<<  >>          Bitwise left and right shifts
&               Bitwise AND
^               Bitwise XOR
|               Bitwise OR
==  !=  >  >=  <  <=  is  is not  in  not in     Comparisons, identity, and membership operators
not             Logical NOT
and             AND
or              OR
```

If two operators have the same precedence, the expression is evaluated from left to right.