Operators

Operators are special symbols in Python that carry out arithmetic or logical computation. The value that the operator operates on is called the operand.

Operator Types

1. Arithmetic operators
2. Comparison (Relational) operators
3. Logical (Boolean) operators
4. Assignment operators
5. Special operators

Arithmetic Operators

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication etc.

```
+ , -, *, /, %, //, **  are arithmetic operators
```

Example:

In [3]:

```
x, y = 21, 4

#addition
print(x + y)
#subtraction(-)
print(x - y)
#multiplication(*)
print(x * y)
#division(/)
print(x / y)
```

```
25
17
84
5.25
```

In [4]:

```
#modulo division (%)
print(x % y)
#Floor Division (//)
print(x // y)
#Exponent (**)
print(x ** y)
```

```
1
5
194481
```

In [ ]:

```
Precedence of operators

()  Brackets


**  Exponentiation (raise to the power)

* / % //    Multiply, divide, modulo and floor division


+ - Addition and subtraction

<= < > >=   Comparison operators

<> == !=    Equality operators

= %= /= //= -= += *= **=    Assignment operators

is is not   Identity operators

in not in   Membership operators

not or and  Logical operators
```

In [ ]:

```
x=(8+2)/5
x
```

Out[3]:

```
2.0
```

In [ ]:

```
y=8+2/5
y
```

Out[5]:

```
8.4
```

In [ ]:

```
#ASSIGNMENT OPERATOR
a=10
print(a)
#a=a+1
a+=1
print(a)
```

```
10
11
```

Relational (Comparision) Operators

Comparison operators are used to compare values. It either returns True or False according to the condition.

```
>, <, ==, !=, >=, <= are comparision operators
```

In [6]:

```
a, b = 10, 20
#check a is less than b
a < b
```

Out[6]:

```
True
```

In [ ]:

```
#check a is greater than b
a > b
```

Out[5]:

```
False
```

In [7]:

```
#check a is equal to b
a == b
```

Out[7]:

```
False
```

In [ ]:

```
#check a is not equal to b (!=)
a!=b
```

Out[7]:

```
True
```

In [ ]:

```
#check a greater than or equal to b
a>=b
```

Out[9]:

```
False
```

In [ ]:

```
#check a less than or equal to b
a<=b
```

Out[10]:

```
True
```

In [8]:

```
#A
10<77
```

Out[8]:

```
True
```

In [9]:

```
#B
a<6
```

Out[9]:

```
False
```

In [10]:

```
#C
b==10+10
```

Out[10]:

```
True
```

In [ ]:

```
#D
a+b<100
```

In [ ]:

```
a*3>=b+10
```

In [ ]:

```
#create atlest 5 statments involving relational operators
```

Logical Operators

Logical operators are **and, or, not** operators.

In [ ]:

```
a, b = 10,20

#and operator - Returns True if both statements are true
a==10 and b==20
```

Out[8]:

```
True
```

In [ ]:

```
a!=10 and b==20
```

In [ ]:

```
a>5 and b<20
```

In [11]:

```
name='nilay'
name=='Nilay'
```

Out[11]:

```
False
```

In [12]:

```
height=174.5
height>100.5 and name=='nilay'
```

Out[12]:

```
True
```

In [13]:

```
# or operator - Returns True if one of the statements is true

a!=10 and b==20
```

Out[13]:

```
False
```

In [14]:

```
a!=10 or b==20
```

Out[14]:

```
True
```

In [ ]:

```

```

In [18]:

```
# not operator - Reverse the result, returns False if the result is true and vice versa

a==10
```

Out[18]:

```
True
```

In [19]:

```
not a==10
```

Out[19]:

```
False
```

In [ ]:

```
not False
```

In [ ]:

```
x==10 and y==20
# T   and   T
```

In [ ]:

```
x+20>=y+10 or 30+y<50
#30>=30       50<50
#   T      or F
```

In [ ]:

```
#create atlest 5 statments involving relational & logical operators
```

In [ ]:

```
per=66
(per>=60) and (per<=75)
    #a      and     b
```

Out[7]:

```
True
```

Assignment operators

Assignment operators are used in Python to assign values to variables.

a = 5 is a simple assignment operator that assigns the value 5 on the right to the variable a on the left.

```
=,  +=,  -=,  *=,  /=,  %=,  //=,  **=, &=,  |=,  ^=,  >>=,  <<= are Assignment op
erators
```

In [ ]:

```
a=10
print(a)
#a=a+1
a+=1
print(a)
```

```
10
11
```

In [ ]:

```
print(a)
#a=a*2
a*=2
print(a)
```

```
11
22
```

In [ ]:

```
a = 10

a += 10         #add AND
print(a)

#subtract AND (-=)

#Multiply AND (*=)

#Divide AND (/=)

#Modulus AND (%=)

#Floor Division (//=)

#Exponent AND (**=)
```

```
20
```

In [ ]:

```
#multiple variable init.
x,y,z=2,4,6
x,y,z
```

In [ ]:

```
"""



"""
```

Special Operators

MemberShip Operators

in and not in are the membership operators in Python.

They are used to test whether a value or variable is found in a sequence (string, list, tuple, set and dictionary).

In [ ]:

```
lst = [1, 2, 3, 4]
print(11 not in lst)       #check 1 is present in a given list or not

#check 5 is present in a given list
```

```
True
```

In [ ]:

```
d = {1: "a", 2: "b"}
print(1 in d)
print("a" in d)
```

```
True
False
```