Ques. What is the operator?

Arithmetic Operators

Operators	Descrption	Result
Addition(+)	Adds the values on either side of the operator.	3+4=7
Subtraction(-)	Subtracts the value on the right from the one on the left.	3+4=-1
Multiplication(*)	Multiplies the values on either side of the operator.	3*4=12
Division(/)	Divides the value on the left by the one on the right. Notice that division results in a floating-point value.	3/4=0.75
Exponentiation(**)	Raises the first number to the power of the second.	3**4=81
Floor Division(//)	Divides and returns the integer value of the quotient. It dumps the digits after the decimal.	10//3=3
Modulus(%)	Divides and returns the value of the remainder.	3%4=3

Comparison(Relational) Operator

Operators	Descrption	Result
Less than(<)	This operator checks if the value on the left of the operator is lesser than the one on the right.	3<4=True
Greater than(>)	It checks if the value on the left of the operator is greater than the one on the right.	3>4=False
Less than or equal to(<=)	It checks if the value on the left of the operator is lesser than or equal to the one on the right.	7<=7 = True
Greater than or equal to(>=)	It checks if the value on the left of the operator is greater than or equal to the one on the right.	0>=0 = True

Operators	Descrption	Result
Equal to(==)	This operator checks if the value on the left of the operator is equal to the one on the right.(1 is equal to the Boolean value True, but 2 isn't. Also, 0 is equal to False.)	3==3.0 = True 1==True = True 7==True = False 0==False = True 0.5==True = False

Assignment Operators

Operators	Descrption	Result

Assign(=)	Assigns a value to the expression on the left. Notice that = = is used for comparing, but = is used for assigning.	>>> a=7
		>>> print(a)
		//output:- 7

Add and Assign(+=)	Adds the values on either side and assigns it to the expression on the left. a+=10 is the same as a=a+10.	>>> a+=2
>>> print(a) //output:- 9		

Divide and Assign(/=)	Divides the value on the left by the one on the right. Then it assigns it to the expression on the left.	>>> a/=7
		>>> print(a)
		//output:- 1.0

Multiply and Assign(*=)	Multiplies the values on either sides. Then it assigns it to the expression on the left.	>>> a*=8
		>>> print(a)
		// 8.0

Operators	Descrption	Result
Modulus and Assign(%=)	Performs modulus on the values on either side. Then it assigns it to the expression on the left.	>>> a%=3 >>> print(a) //output:-2.0
Exponent and Assign(**=)	Performs exponentiation on the values on either side. Then assigns it to the expression on the left.	>>> a**=5 >>> print(a) //output:-32.0
Floor-Divide and Assign(//=)	Performs floor-division on the values on either side. Then assigns it to the expression on the left.	>>> a//=3 >>> print(a) //output:-10.0

Bitwise Operators

Operators	Descrption	Result
Binary AND(&)	It performs bit by bit AND operation on the two values. Here, binary for 2 is 10, and that for 3 is 11. &-ing them results in 10, which is binary for 2.	>>> 2&3 //output:-2
Binary OR(	)	--
Binary XOR(^)	--	--
Binary One's Complement(~)	--	--
Binary Left-Shift(<<)	--	--
Binary Right-Shift(>>)	--	--

Ques. What is membership operator and identity operators?

- **Membership Operators:-** These operators help validate whether a given element is present in or is a member of the given sequence of data. This sequence of data can be a list, string or a tuple
  There are 2 types of Membership operater

1. in Operator:

```
lst1 = ['Ajay', 'Bobby','Ashok', 'Vijay', 'Anil', 'Rahul','Alex', 'Christopher']
if 'Ajay' in lst1:
    print('Name Ajay exists in lst1')

Output:- Name Ajay exists in lst1
```

2. not in Operator:

```
lst1 = ['Ajay', 'Bobby','Ashok', 'Vijay', 'Anil', 'Rahul','Alex', 'Christopher']
   if 'Raghav' not in lst1:
   print ('Name Raghav does not exists in lst1')

output:- Name Raghav exists in lst1
```

- **Identity Operators:-** These operators help in determining whether a value belongs to a certain class or a certain type, i.e they help in determining the identity of the object. It is useful in finding out the data type a variable holds.

1. is Operator:

```
a = 'London'
b = 'London'

if a is b: print ('a is b')
else: print ('a is not b')

if a is c: print('a is c')
else: print ('a is not c')

Output:-
a is b
a is not c
```

2. is not Operator:

```
x = ["apple", "banana"]
y = ["apple", "banana"]
z = x

print(x is not z)
print(x is not y)
print(x != y)

Output:-
False
```

```
True
False
```

Ques. Python Variables?

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)

```
#Legal variable names:
myvar = "John"
my_var = "John"
_my_var = "John"
myVar = "John"
MYVAR = "John"
myvar2 = "John"

#Illegal variable names:
2myvar = "John"
my-var = "John"
my var = "John"
```

Example

```
x = 5
y = "Mohit"
print(x)
print(y)
```

output:- 5
Mohit

```
# double quotes are the same as single quotes:
x = 4 # x is of type int
x = "saxena" # x is now of type str
x = 'mohit' # x is now of type str
print(x)

Output:-Mohit
```

```
x, y, z = "Orange", "Banana", "Cherry"
   print(x)
   print(y)
```

```
    print(z)

output:- Orange<br>Banana<br>Cherry

# Assign Value to Multiple Variables
x = y = z = "Orange"
print(x)
print(y)
print(z)

Output:-
Orange
Orange
Orange
```

Output Variables(combine both text and a variable)

```
x = "awesome"
print("Python is " + x)

output:- Python is awesome

# Example 2
-----------
x = "Python is "
y = "awesome"
z =  x + y
print(z)

output:-Python is awesome
```

```
x = 5
y = 10
print(x + y)
```

output:- 15

Note:- If you try to combine a string and a number, Python will give you an error:

```
x = 5
y = "John"
print(x + y)
```

output:- TypeError: unsupported operand type(s) for +: 'int' and 'str'

Ques. Global Variables?

- Variables that are created outside of a function.
- Global variables can be used by everyone, both inside of functions and outside. **Example:-**

```
x = "awesome"
def myfunc():
  print("Python is " + x)
myfunc()

output:- Python is awesome
```

```
x = "awesome"
def myfunc():
  x = "fantastic"
  print("Python is " + x)
myfunc()
print("Python is " + x)

output:- Python is fantastic<br>Python is awesome
```