Python Operators And Operand

Touhiduzzaman Pavel

Presenter

Touhiduzzaman Pavel

Junior Instructor (Tech) Computer

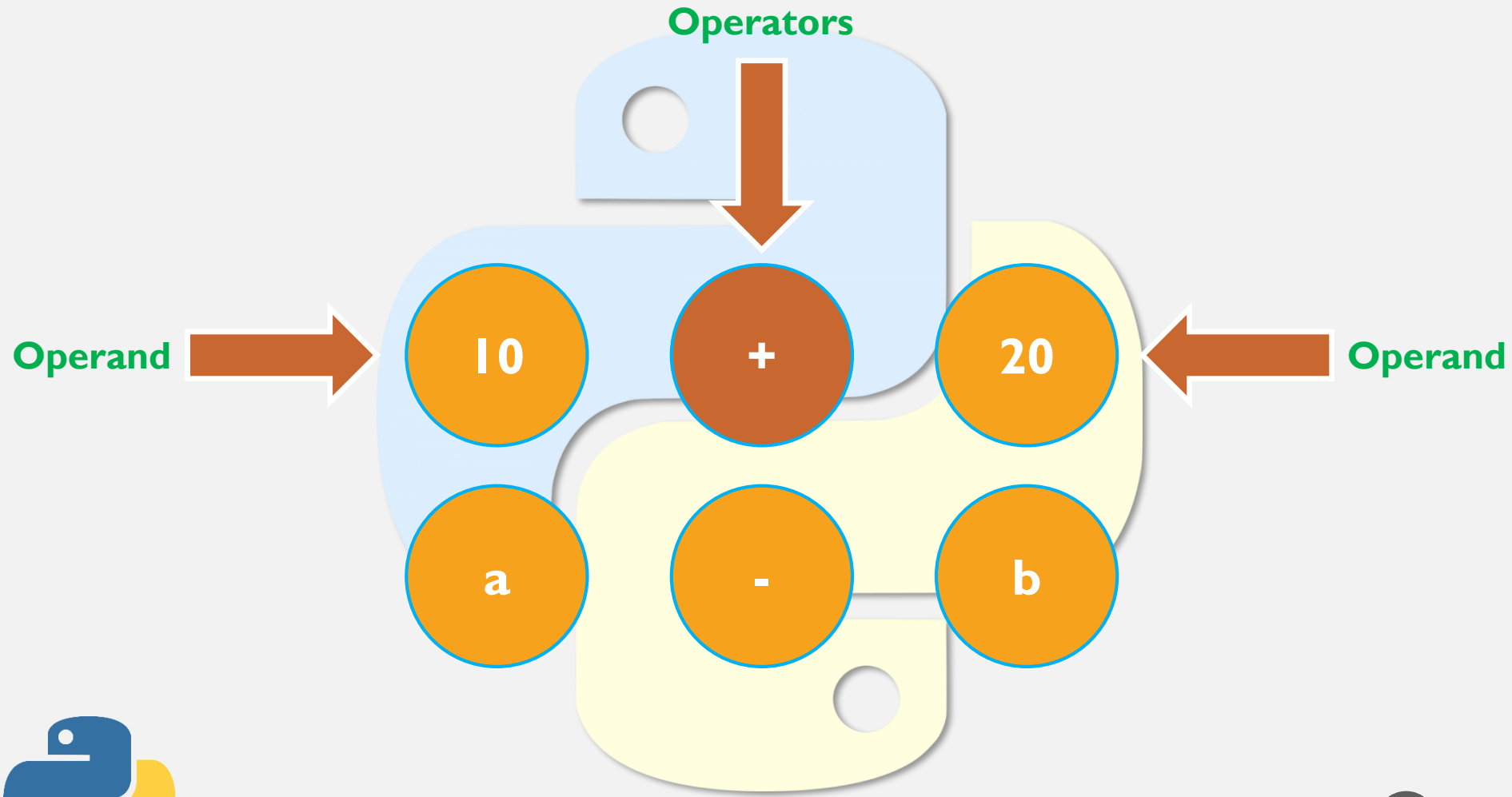
Rajshahi Mohila Polytechnic Institute

Cell >> 01738-276055

WHAT IS OPERATORS AND OPERANDS

- An operator, in computer programming, is a symbol that usually represents an action or process.
- It is a symbol that perform operation on one or more operands.
- An operand is a value that a given operator is applied to.
- Operators required operands to perform their job.

EXAMPLE.....



TYPES OF OPERATOR

Python language supports the following types of operators.

- Arithmetic Operators
- Comparison / Relational Operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

ARITHMETIC OPERATORS

Arithmetic operators take numerical values as their operands and return a single numerical value.

Here a = 10 b =20

Operator	Description	Example
+ Addition	Adds values on either side of the operator.	a + b = 30
- Subtraction	Subtracts right hand operand from left hand operand.	a – b = -10
* Multiplication	Multiplies values on either side of the operator	a * b = 200
/ Division	Divides left hand operand by right hand operand	b / a =2
% Modulus	Divides left hand operand by right hand operand and returns remainder	b % a = 0
** Exponent	Performs exponential power calculation on operators	a**b =10 to the power 20
// Floor Division	The division of operands where the result is the quotient in which the digits after the decimal point are removed.	9//2 = 4 9.0//2.0 = 4.0

COMPARISON OPERATORS

These operators compare the values on either sides of them and decide the relation among them. They are also called Relational operators.

Here a = 10 b =20

Operator	Description	Example
== Equal to	If the values of two operands are equal, then the condition becomes true	a == b is not true.
!= Not Equal to	If values of two operands are not equal, then condition becomes true.	
<> Not Equal to	If values of two operands are not equal, then condition becomes true.	a <> b is true. This is similar to != operator.
> Greater than	If the value of left operand is greater than the value of right operand, then condition becomes true.	a > b is not true.
< Less than	If the value of left operand is less than the value of right operand, then condition becomes true.	a < b is true.
>= Greater than or equal to	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	a >= b is not true.
<= Less than or equal to	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	a <= b is true.

ASSIGNMENT OPERATORS

Assignment operators are used to assign values to variables.

Here a = 10 b =20

Operator	Description	Example
= Equal AND	Assigns values from right side operands to left side operand	c = a + b assigns value of a + b into c
+= Add AND	It adds right operand to the left operand and assign the result to left operand	c += a is equivalent to c = c + a
-= Subtract AND	It subtracts right operand from the left operand and assign the result to left operand	c -= a is equivalent to c = c - a
*= Multiply AND	It multiplies right operand with the left operand and assign the result to left operand	c *= a is equivalent to c = c * a
/= Divide AND	It divides left operand with the right operand and assign the result to left operand	c /= a is equivalent to c = c / ac /= a is equivalent to c = c / a

PYTHON BITWISE OPERATORS

Bitwise operator works on bits and performs bit by bit operation. **Assume if a = 60; and b = 13; Now in binary format they will be as follows –**

a = 0011 1100

b = 0000 1101

----

a&b = 0000 1100

a|b = 0011 1101

a^b = 0011 0001

~a = 1100 0011

PYTHON BITWISE OPERATORS

Assume if a = 60; and b = 13; Now in binary format they will be as follows −
a = 0011 1100 b = 0000 1101

Operator	Description	Example
& Binary AND	Operator copies a bit to the result if it exists in both operands	a & b means 00001100
| Binary OR	It copies a bit if it exists in either operand	a|b = 61 means 00111101
^ Binary XOR	It copies the bit if it is set in one operand but not both.	a^b = 49 means 00110001
~ Binary Ones Complement	It is unary and has the effect of 'flipping' bits.	a = -61 (means 1100 0011 in 2's complement form due to a signed binary number.
<< Binary Left Shift	The left operands value is moved left by the number of bits specified by the right operand.	a << = 240 means11110000
>> Binary Right Shift	The left operands value is moved right by the number of bits specified by the right operand.	a >> = 15 means 00001111

PYTHON LOGICAL OPERATORS

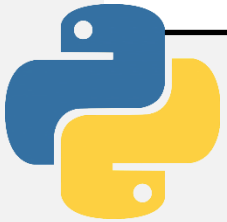
Logical operators are operations between two logical expressions.
Assume variable a holds 10 and variable b holds 20 then

Operator	Description	Example
and Logical AND	If both the operands are true then condition becomes true.	(a and b) is true.
or Logical OR	If any of the two operands are non-zero then condition becomes true.	(a or b) is true.
not Logical NOT	Used to reverse the logical state of its operand.	Not(a and b) is false.

PYTHON MEMBERSHIP OPERATORS

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

Operator	Description	Example
in	Evaluates to true if it finds a variable in the specified sequence and false otherwise.	x in y, here in results in a 1 if x is a member of sequence y. a = "Hello" print("H" in a) >> True print("T" in a) >> False
not in	Evaluates to true if it does not finds a variable in the specified sequence and false otherwise.	x not in y, here not in results in a 1 if x is not a member of sequence y. a = "Hello" print("Q" not in a) >> True



PYTHON IDENTITY OPERATORS

Identity operators compare the memory locations of two objects.

- ➥ Every variable in python is an object.
- ➥ Objects are dynamically created.
- ➥ Objects do not have name but references.

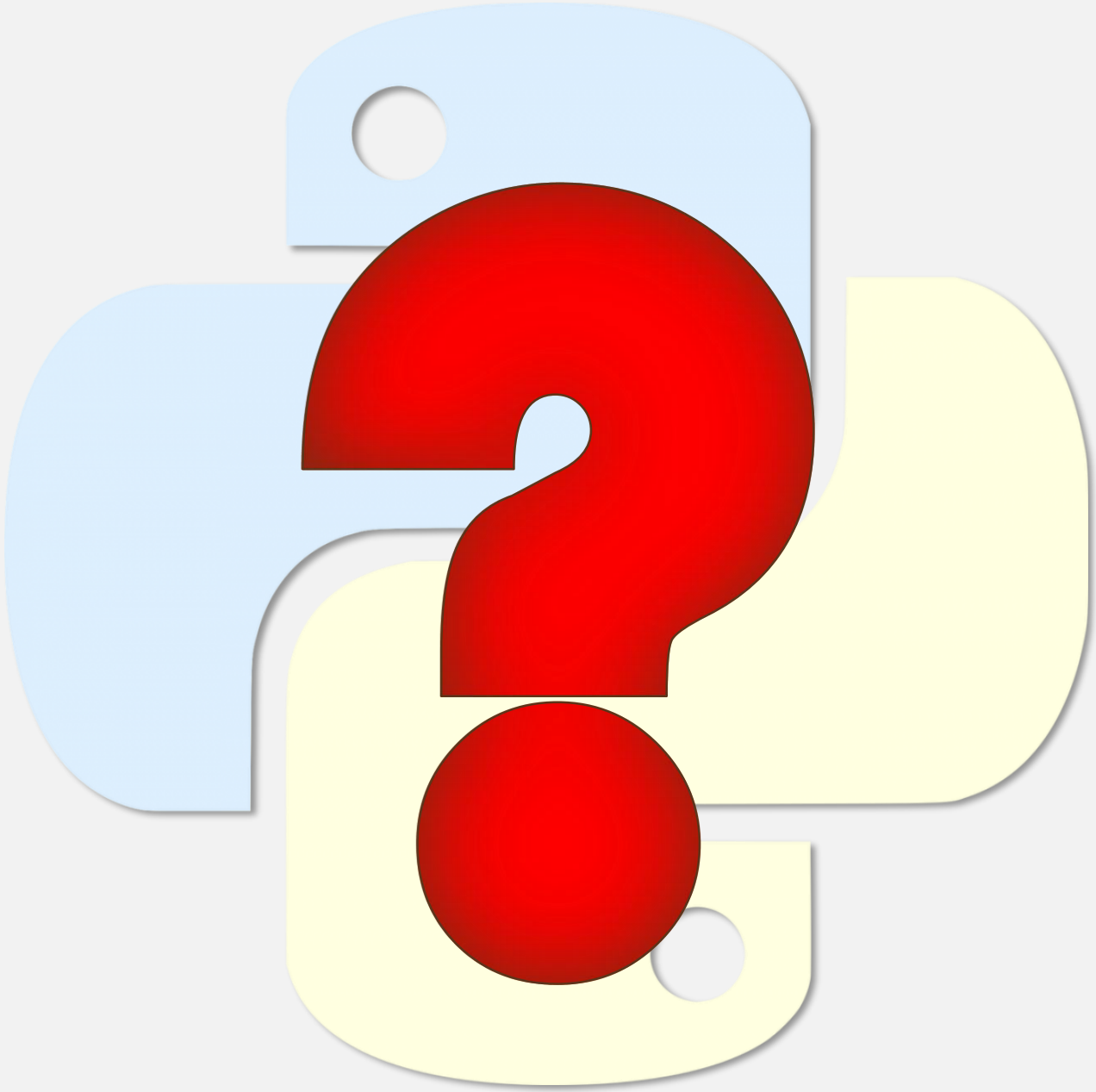
There are two Identity operators explained below:

Operator	Description	Example
is	Evaluates to true if the variables on either side of the operator point to the same object and false otherwise.	x is y, here is results in 1 if idx equals id y. a = "Hi" b = "Hi" print(a is b) >> True
is not	Evaluates to false if the variables on either side of the operator point to the same object and true otherwise.	x is not y, here is not results in 1 if idx is not equal to idy. a = 5 b = 5 print(a is not b) >> False

PYTHON OPERATORS PRECEDENCE

An **operator precedence** is the order that an operator is executed.

Sl No	Operator & Description
1	** Exponentiation (raise to the power)
2	~ + - Complement, unary plus and minus (method names for the last two are +@ and -@)
3	* / % // Multiply, divide, modulo and floor division
4	+ - Addition and Subtraction
5	>> << Right and left bitwise shift
6	& Bitwise 'AND'
7	^ | Bitwise exclusive `OR' and regular `OR'
8	<= < > >= Comparison operators
9	<> == != Equality operators
10	= %= /= //= -= += *= **= Assignment operators
11	is is not Identity operators
12	in not in Membership operators
13	not or and Logical operators



Stay Home
Stay Safe
Happy learning


THANK
YOU