Python Programming

KUMARAGURU
college of technology
character is life

KCED
Learning Augmented

Operators and Expressions

Ms. Aswini D

Department of Computer Science and Engineering

Contents

- Operators
- Types of Operators
- Operator Precedence
- Associativity

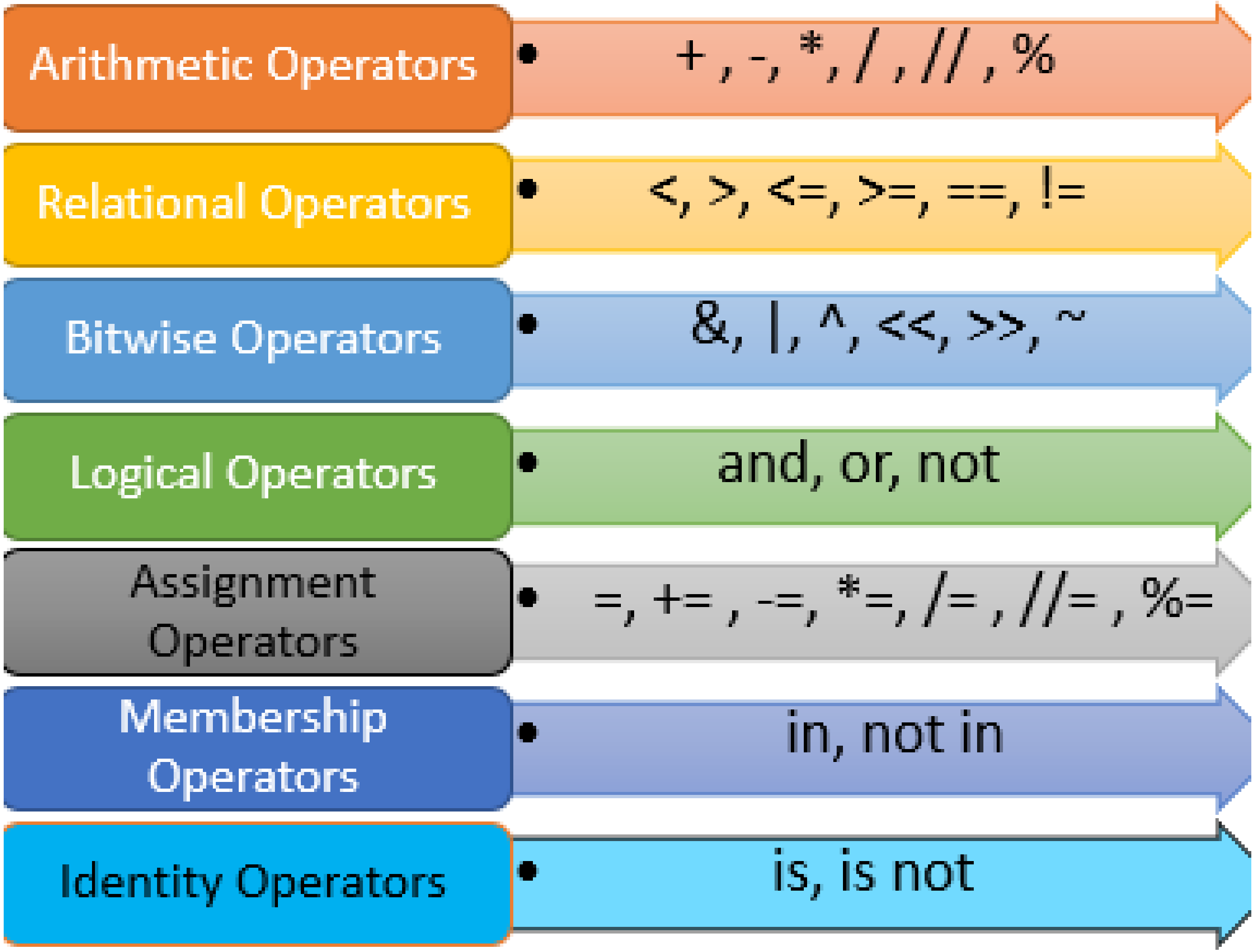
Introduction

- Operators are special symbols in Python that carry out various computation on operands.
- Python supports different kind of operators such as Arithmetic, Relational, Logical, etc.,
- In Python program, most of the statements contain expression.
- An expression is a statement with operators and operands which produces result upon evaluation.
- Operators are symbols which specifies the operation to be performed on values or variables.
- These values or variables are known as operands.

Types of Operators

- Operators are used to perform arithmetic and logical operation.
- It is classified into following types.
  - Arithmetic Operators
  - Relational Operators
  - Logical Operators
  - Bitwise Operators
  - Assignment Operators
  - Membership operators
  - Identity Operators

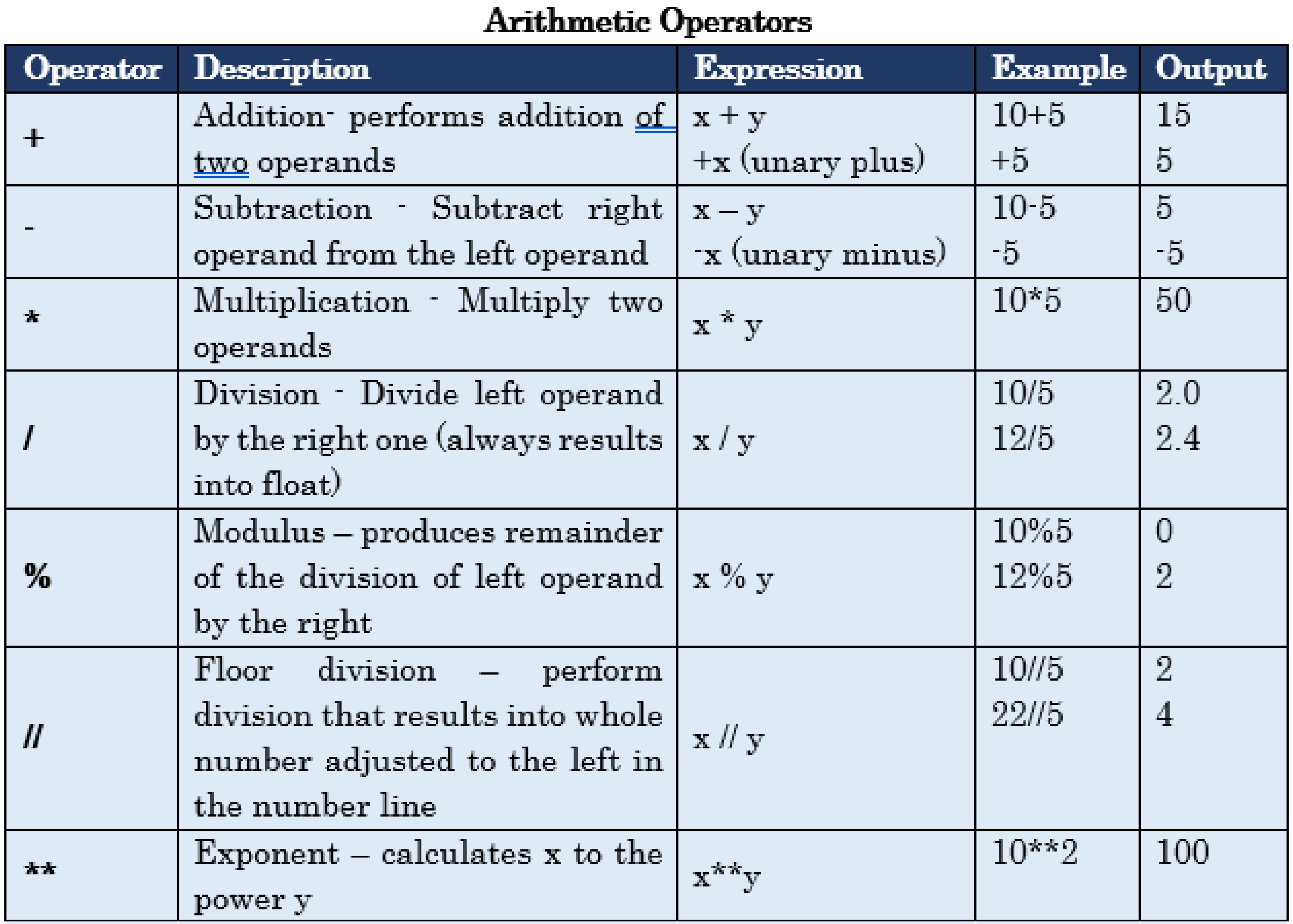
Category of operators supported in Python



Arithmetic Operators	• + , -, *, / , // , %
Relational Operators	• <, >, <=, >=, ==, !=
Bitwise Operators	• &, |, ^, <<, >>, ~
Logical Operators	• and, or, not
Assignment Operators	• =, += , -=, *=, /= , //= , %=
Membership Operators	• in, not in
Identity Operators	• is, is not

Arithmetic Operators

- Arithmetic operators are used to perform arithmetic operations such as addition, subtraction, multiplication and division.



Arithmetic Operators

Operator	Description	Expression	Example	Output
+	Addition- performs addition of two operands	x + y +x (unary plus)	10+5 +5	15 5
-	Subtraction - Subtract right operand from the left operand	x – y -x (unary minus)	10-5 -5	5 -5
*	Multiplication - Multiply two operands	x * y	10*5	50
/	Division - Divide left operand by the right one (always results into float)	x / y	10/5 12/5	2.0 2.4
%	Modulus – produces remainder of the division of left operand by the right	x % y	10%5 12%5	0 2
//	Floor division – perform division that results into whole number adjusted to the left in the number line	x // y	10//5 22//5	2 4
**	Exponent – calculates x to the power y	x**y	10**2	100

Example Program

Write a Python program to find the profit of a product. Read the cost and selling price of a product as input.

```
#To find profit of a product
cost=float(input("Enter the Cost of a Product :")) # read cost
# read selling price
selling_price=float(input("Enter the Selling price of a Product :"))
profit=selling_price-cost # calculate the profit
print("Profit Earned = ", profit) # display the result
```

Output

```
Enter the Cost of a Product :1200
Enter the Selling price of a Product
:1500
Profit Earned = 300.0
```

Example - Program

Write a program to convert the temperature from Celsius to Fahrenheit.

```
#Program to convert temperature

Celsius=float(input(“Enter the temperature in Celsius:”))

Fahrenheit =(Celsius*1.8)+32

print(“Temperature in Fahrenheit is:”, Fahrenheit)
```

```
Output:
Enter the temperature in Celsius :28
Temperature in Fahrenheit is: 82.4
```

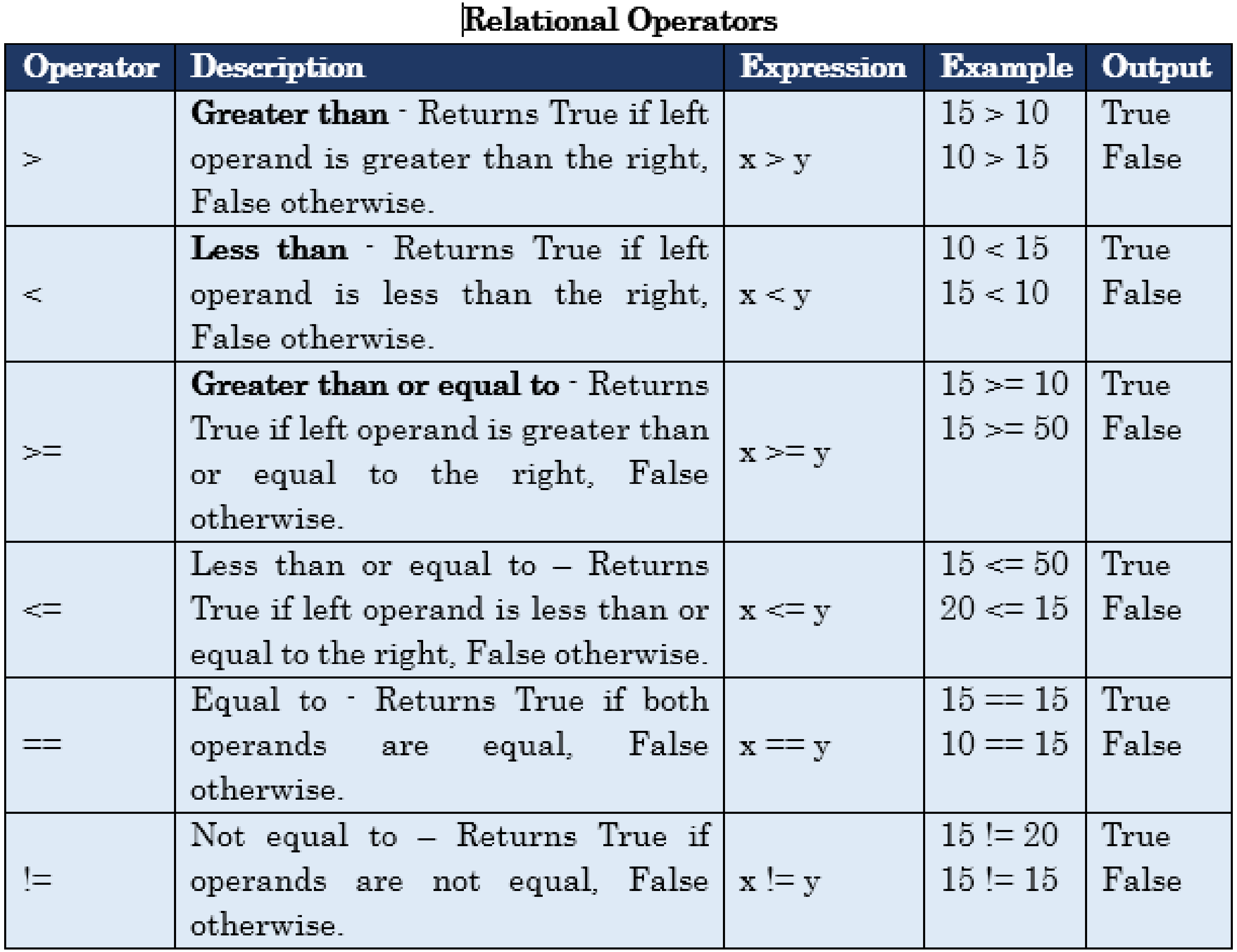
Try it out!!!

Write a Python program to calculate Simple Interest. Read the principle(p), rate of interest(r) and number of years(n) from the user.

Formula for calculation simple interest is SI=pnr/100.

Relational Operators

- Relational operators are used to compare the values.
- It is also called as comparison operators.
- It always returns Boolean value as a result, i.e. True or False.

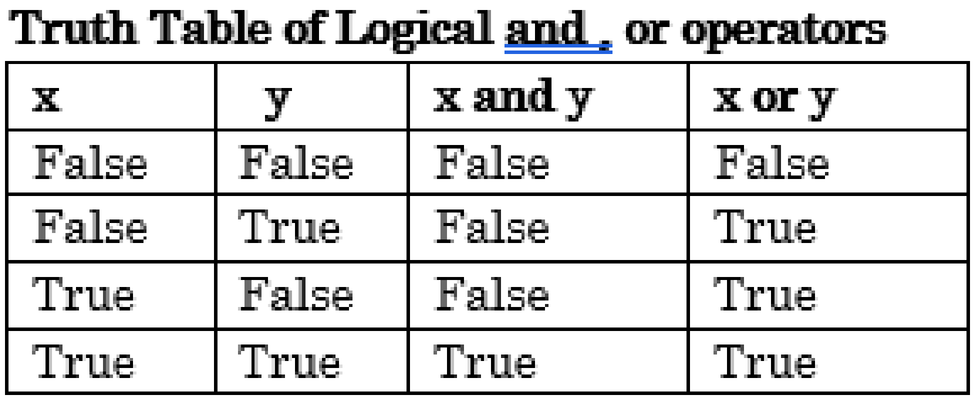


Relational Operators

Operator	Description	Expression	Example	Output
>	**Greater than** - Returns True if left operand is greater than the right, False otherwise.	x > y	15 > 10 10 > 15	True False
<	**Less than** - Returns True if left operand is less than the right, False otherwise.	x < y	10 < 15 15 < 10	True False
>=	**Greater than or equal to** - Returns True if left operand is greater than or equal to the right, False otherwise.	x >= y	15 >= 10 15 >= 50	True False
<=	Less than or equal to – Returns True if left operand is less than or equal to the right, False otherwise.	x <= y	15 <= 50 20 <= 15	True False
==	Equal to - Returns True if both operands are equal, False otherwise.	x == y	15 == 15 10 == 15	True False
!=	Not equal to – Returns True if operands are not equal, False otherwise.	x != y	15 != 20 15 != 15	True False

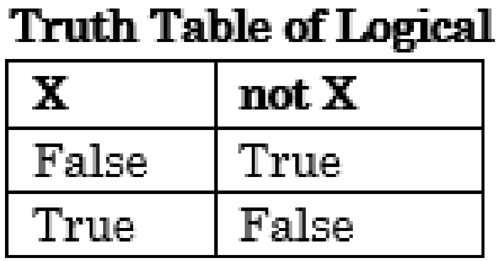
Logical Operators

- Logical operators performs logical **AND, OR and NOT** on operands.
- If an operand contains any non-zero values, then it is considered as True value, zero is considered as False value.
- logical operators are used along with conditional statements



Truth Table of Logical and , or operators

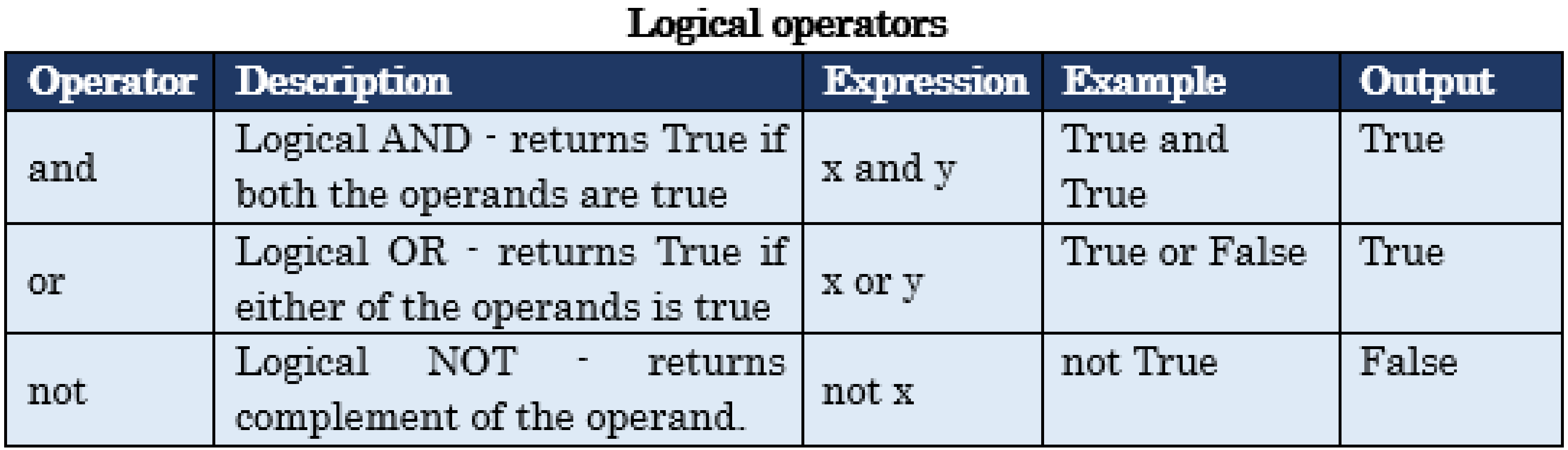
x	y	x and y	x or y
False	False	False	False
False	True	False	True
True	False	False	True
True	True	True	True



Truth Table of Logical not operator

X	not X
False	True
True	False

Logical operators

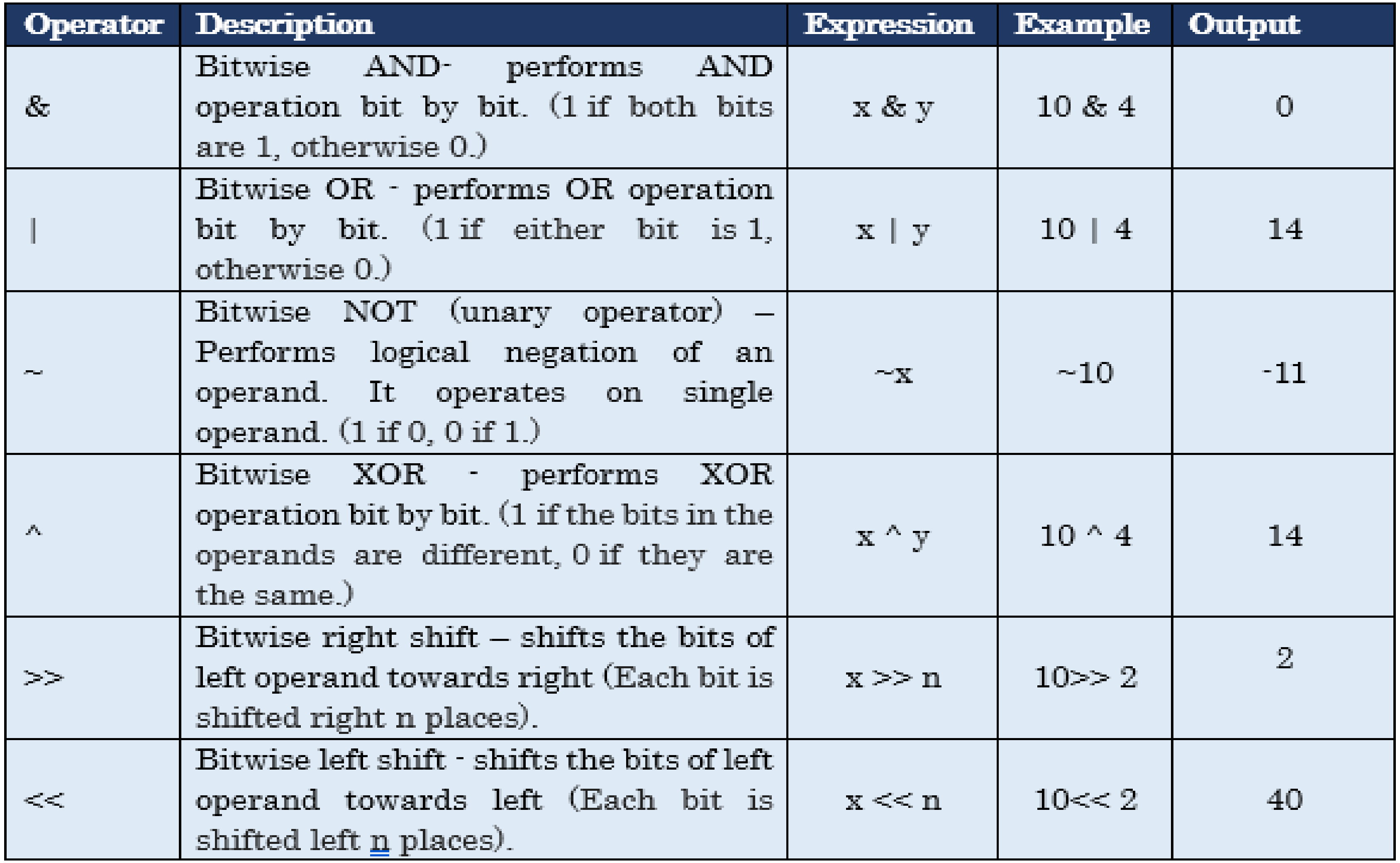


Logical operators

Operator	Description	Expression	Example	Output
and	Logical AND - returns True if both the operands are true	x and y	True and True	True
or	Logical OR - returns True if either of the operands is true	x or y	True or False	True
not	Logical NOT - returns complement of the operand.	not x	not True	False

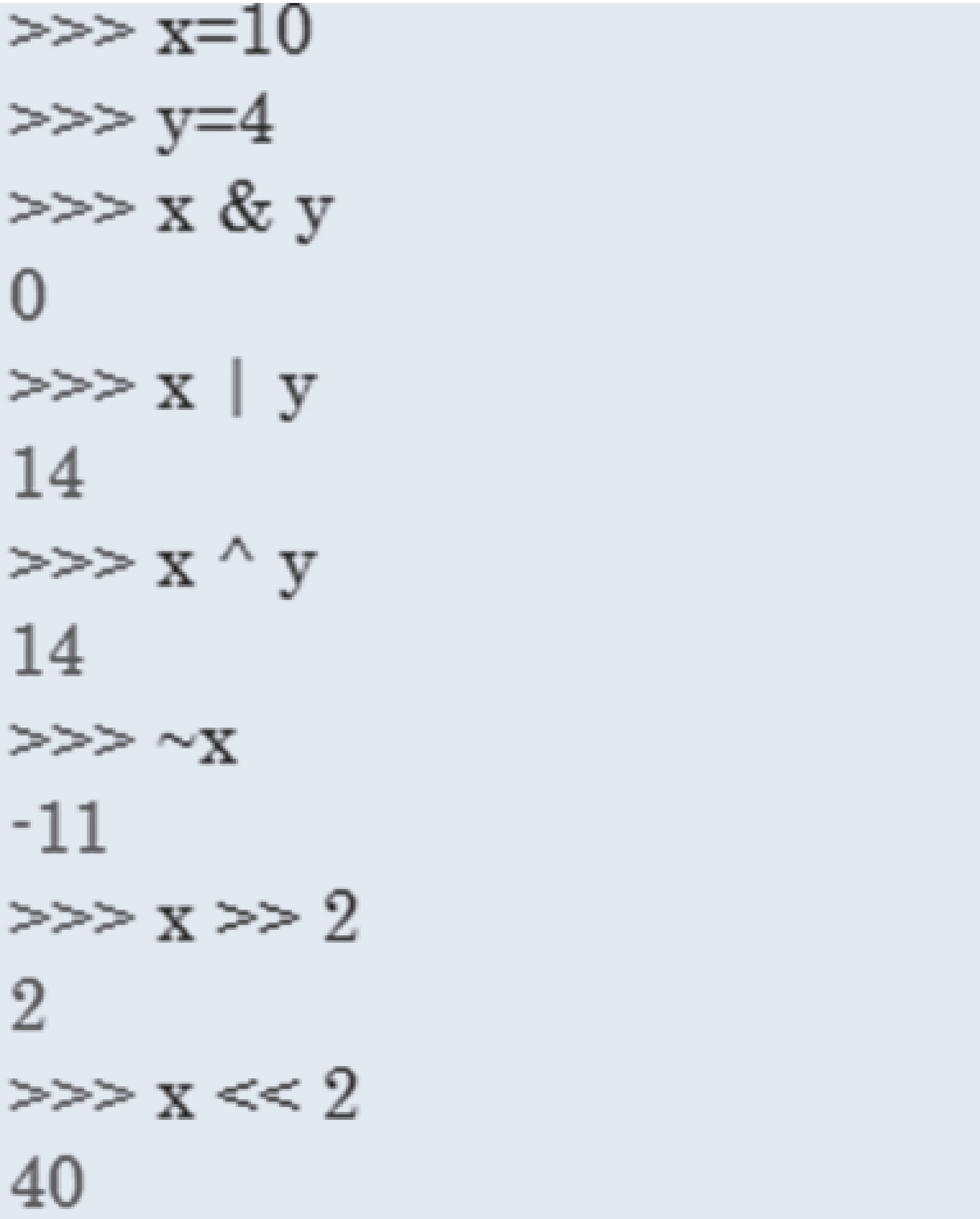
Bitwise operators

Bitwise operators



Operator	Description	Expression	Example	Output
&	Bitwise AND- performs AND operation bit by bit. (1 if both bits are 1, otherwise 0.)	x & y	10 & 4	0
|	Bitwise OR - performs OR operation bit by bit. (1 if either bit is 1, otherwise 0.)	x | y	10 | 4	14
~	Bitwise NOT (unary operator) – Performs logical negation of an operand. It operates on single operand. (1 if 0, 0 if 1.)	~x	~10	-11
^	Bitwise XOR - performs XOR operation bit by bit. (1 if the bits in the operands are different, 0 if they are the same.)	x ^ y	10 ^ 4	14
>>	Bitwise right shift – shifts the bits of left operand towards right (Each bit is shifted right n places).	x >> n	10>> 2	2
<<	Bitwise left shift - shifts the bits of left operand towards left (Each bit is shifted left n places).	x << n	10<< 2	40

- Bitwise operators implicitly converts operands as sequences of binary digits (0 & 1).
- Performs bit level operation on operands and returns the result in decimal.

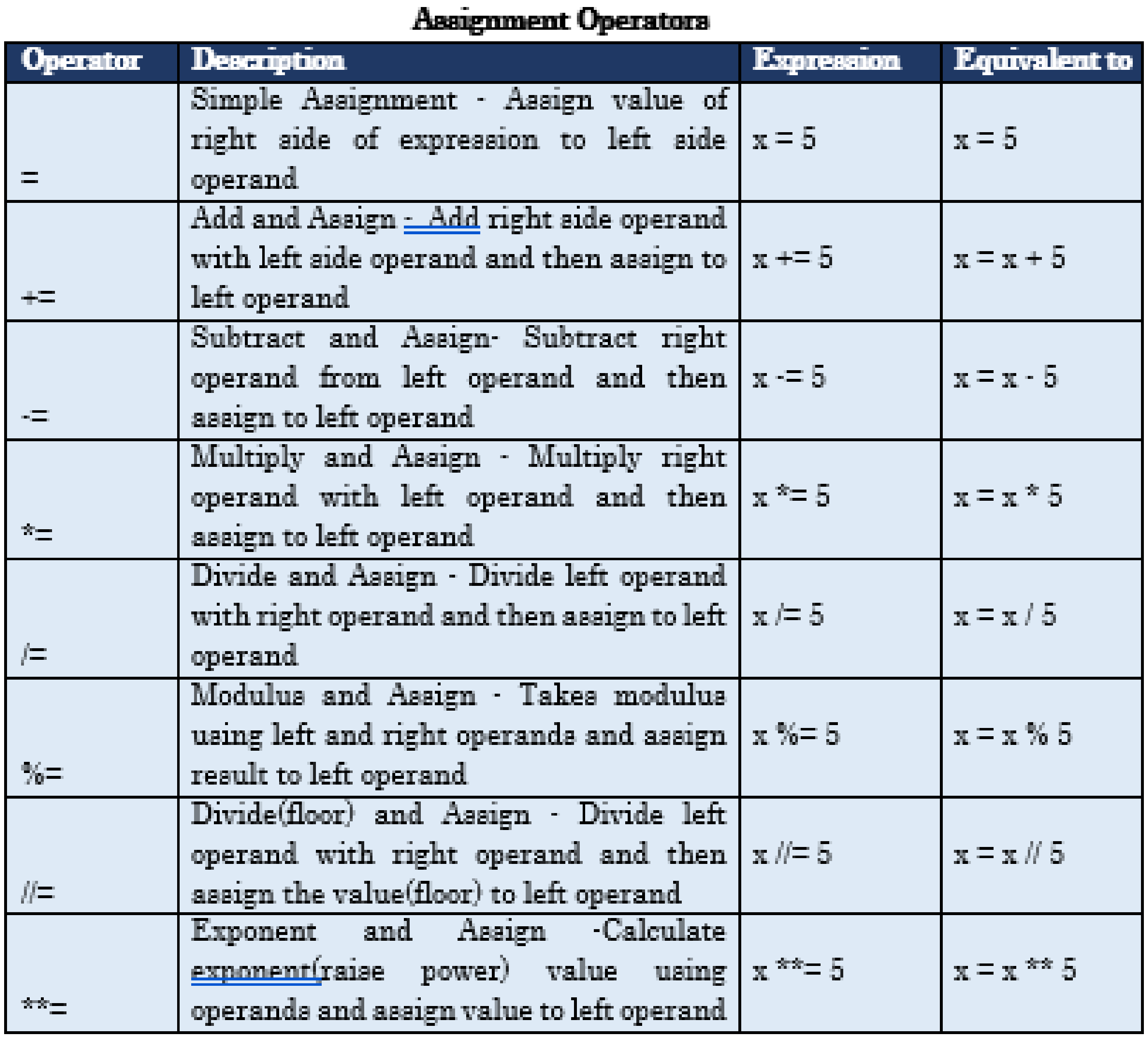


```
>>> x=10
>>> y=4
>>> x & y
0
>>> x | y
14
>>> x ^ y
14
>>> ~x
-11
>>> x >> 2
2
>>> x << 2
40
```

Assignment operators

- Assignment operators are used to assign values to variables.
- It assigns right hand side operand to left hand side operand of the assignment operator.
- Two types of Assignment operators: **Simple and Compound.**
- **Simple Assignment Operator**
  - x=15 is a simple assignment operator that assigns the value 15 to the variable x on the left.
- **Compound Assignment Operator**
  - x += 15 is the compound assignment operator that adds value 15 to the variable x and later assigns the result to the left operand. It is equivalent to x= x+15.

Assignment operators..ctd



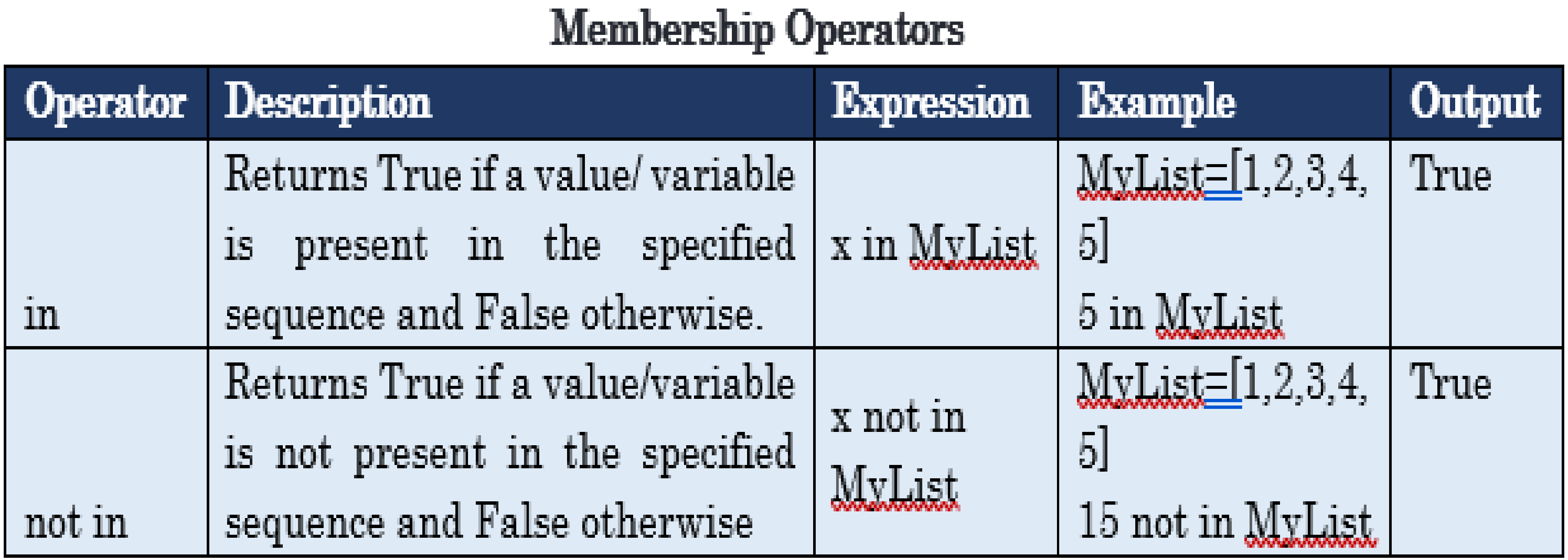
Assignment Operators

Operator	Description	Expression	Equivalent to
=	Simple Assignment - Assign value of right side of expression to left side operand	x = 5	x = 5
+=	Add and Assign - Add right side operand with left side operand and then assign to left operand	x += 5	x = x + 5
-=	Subtract and Assign- Subtract right operand from left operand and then assign to left operand	x -= 5	x = x - 5
*=	Multiply and Assign - Multiply right operand with left operand and then assign to left operand	x *= 5	x = x * 5
/=	Divide and Assign - Divide left operand with right operand and then assign to left operand	x /= 5	x = x / 5
%=	Modulus and Assign - Takes modulus using left and right operands and assign result to left operand	x %= 5	x = x % 5
//=	Divide(floor) and Assign - Divide left operand with right operand and then assign the value(floor) to left operand	x //= 5	x = x // 5
**=	Exponent and Assign -Calculate exponent(raise power) value using operands and assign value to left operand	x **= 5	x = x ** 5

Membership operators

- Python language offers some special type of operators like the membership operators and the identity operators.
- The programming languages like C,C++ ,JAVA does not support these operators.
- Membership operators are used to check whether a value or variable is found in a sequence (string, list, tuple, set and dictionary)

Membership operators..ctd

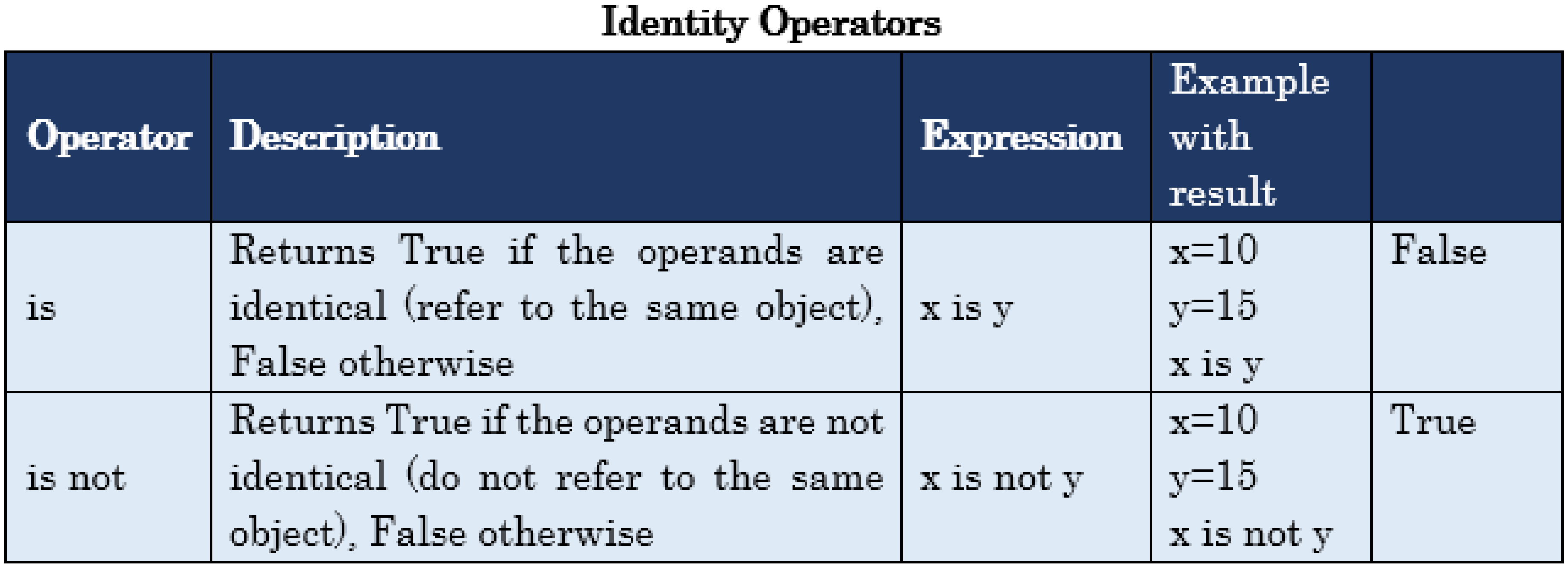


Membership Operators

Operator	Description	Expression	Example	Output
in	Returns True if a value/ variable is present in the specified sequence and False otherwise.	x in MyList	MyList=[1,2,3,4, 5] 5 in MyList	True
not in	Returns True if a value/variable is not present in the specified sequence and False otherwise	x not in MyList	MyList=[1,2,3,4, 5] 15 not in MyList	True

Identity Operators

- Identity operators are used to check whether two operands have same identity i.e. both refers to the same object.
- The identity (address of memory location where the object is stored) of an object can be verified using built-in function id().
- id() returns the address of an operand or object.



Identity Operators

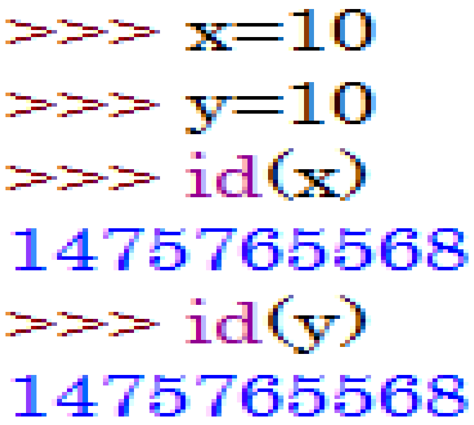
Operator	Description	Expression	Example with result	

is	Returns True if the operands are identical (refer to the same object), False otherwise	x is y	x=10
			y=15
			x is y
				False

is not	Returns True if the operands are not identical (do not refer to the same object), False otherwise	x is not y	x=10
			y=15
			x is not y
				True

Example-Identity operator

Let’s check the address of variable using id() function.



```
>>> x=10
>>> y=10
>>> id(x)
1475765568
>>> id(y)
1475765568
```

Here, x and y refers to the same object, so they are identical.

Expressions

- An expression is a combination of operators and operands that is interpreted to produce some other value.
- An expression is evaluated as per the precedence of its operators
- Example:
  - p = (a + b) >= (c - d)
  - b = (10 + 3) * 4

Operator Precedence

- Python interpreter can evaluate a valid expression.
- There can be more than one operator in an expression.
- To evaluate these type of expressions there is a rule of precedence in Python.
- Operator Precedence guides the order in which operation are carried out.

Operator precedence rule in Python

Operators	Meaning
()	Parentheses
**	Exponent
+, -, ~	Unary plus, Unary minus, Bitwise NOT
*, /, //, %	Multiplication, Division, Floor division, Modulus
+, -	Addition, Subtraction
<<, >>	Bitwise shift operators
&	Bitwise AND
^	Bitwise XOR
|	Bitwise OR
==, !=, >, >=, <, <=, is, is not, in, not in	Comparisons, Identity, Membership operators
not	Logical NOT
and	Logical AND
or	Logical OR

Associativity

- When two operators have the same precedence, associativity helps to determine which the order of operations.
- Associativity is the order in which an expression is evaluated that has multiple operator of the same precedence. Almost all the operators have left-to-right associativity.
- For example, multiplication and floor division have the same precedence. Hence, if both of them are present in an expression, left one is evaluated first.
  - 4*6//2 = 12
- **Exponent operator ** has right-to-left associativity in Python**.

  Ex: 2 **2**3 = 256

Non associative operators

- Some operators like comparison operators do not have associativity in Python.
- There are separate rules for sequences of this kind of operator and cannot be expressed as associativity.
- For example, x < y < z neither means (x < y) < z nor x < (y < z).
- x < y < z is equivalent to x < y and y < z and is evaluates from left-to-right.

Activity

Guess the Output of following expression

Z=4*6+8//2

X= 4/2*2+16/8+5

Try it Out!!!

- Write a program to read the distance between any two cities in kilometer and find the distance in meters, centimeters and miles.
- Write a program to find the Basic Pay, DA, HRA, PF and Net Pay of an employee. Read the wage(salary) and number of days worked as input. (Note HRA, DA and PF are 5%, 15% and 10% of basic pay respectively.)

LEARNING AUGMENTED

Thank You

Tips

Blue light is basically the rays that are emitted by the digital screen, which may cause severe health problems. Flux software for PCs, Twilight app for Smartphones can act as blue light filters.

Micro cosm

The Co-existing Architecture

Kumaraguru Eco Campus

Annually, the **4200 trees** inside the Kumaraguru Campus, on an average, releases **10,92,000 pounds** of oxygen inhaled by **42,000 humans**.

Wellness 360

Find ways to structure and optimize your time for when and where you learn best and keep your **learning on track**

KUMARAGURU
college of technology
character is life

KCED
Learning Augmented