Python-Programming

Module -1

Contents

Introduction to Python.

- **Python** is a very popular general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python is dynamically-typed and garbage-collected programming language.
- Developed by Guido Van Rossum in 1990.
- He wanted the name of his new language short, unique and mysterious
- There are 2 versions of python: python 2 and python3

Advantages of Python

- Readability: maintenance is the most crucial task in programming. Python offers more readability
- Portability: Its Platform independent. Its programs run on all platforms.
- Vast Support of Libraries: It has large collection of in built functionalities known as standard library functions. It supports various third party software like Num Py.(An Extension support for large ,multidimensional arrays and matrices
- Software Integration: It can easily extend, communicate and Integrate with several other languages.
- Developer productivity: There is no need to declare the variables explicitly. also various features which reduce the size of the codes

Features

- High level language
- Simple,readable
- GUI programming support
- Case sensitive
- Platform Independent
- Object oriented programming,functional,procedural
- Easy to debug
- Integrated , Interpreted
- Fewer lines of codes
- No type declaration(dynamic typing)
- Easy to learn, Maintain, and Interactive
- Many built in data types: String, Lists, Tuples, Dictionaries etc

Python IDLE

- IDLE stands for **Integrated Development and Learning Environment.** The lightweight and user-friendly Python IDLE is a tool for Python programming.
- Since version 1.5.2b1, the standard Python implementation has included IDLE, an integrated development environment. Many Linux distributions include it in the Python packaging as an optional component.

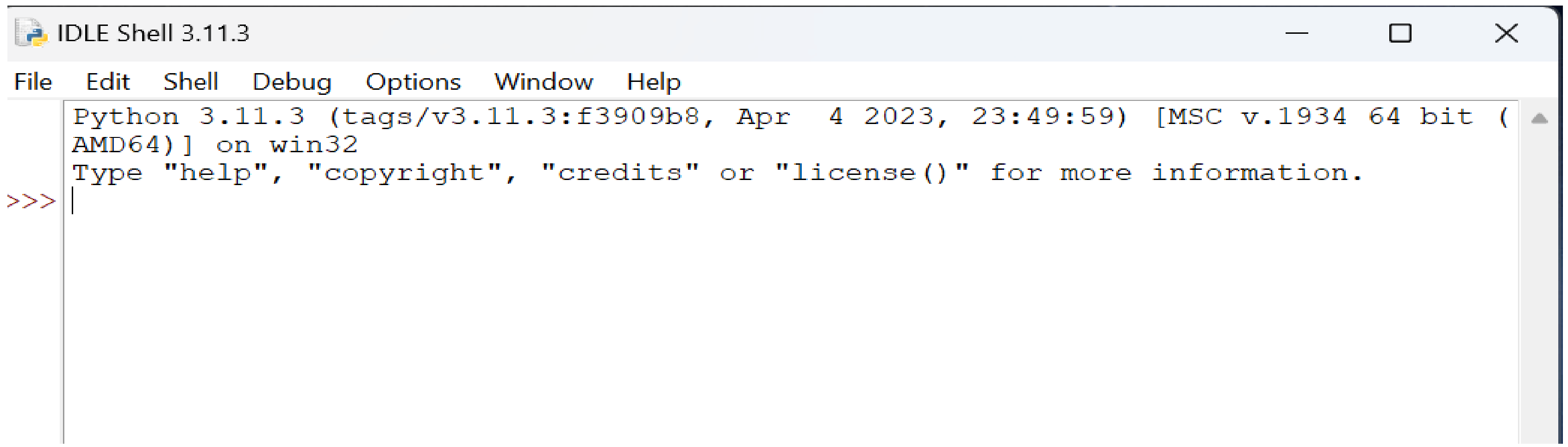
- IDLE is intended to be a simple IDE and suitable for beginners, especially in an educational environment. To that end, it is cross-platform, and avoids feature clutter.

- The python installer in windows contain the IDLE module by default

- IDLE can be used to execute a single line statement just like python shell and also create , modify, and execute python scripts.

- It provides fully featured text editor to create python scripts that include features like syntax highlighting , auto completion and smart indent

- It also have a debugger with stepping and break point features



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
```

An Integrated Development and Learning Environment, sometimes known as IDLE or even IDE, is included with every Python setup. These are a group of programmes that make it easier to write code. Although there are several IDEs available, Python IDLE is fairly basic, making it the ideal tool for a beginner coder.

Python installs for Windows and Mac include Python IDLE. Python IDLE should be easy to locate and download if you run Linux thanks to the package management. After installation, you may use Python IDLE as a file editor or an interactive interpreter.

•Python IDLE is an interactive shell that enables users to easily test and run short bits of Python code without needing to create a whole programme.
•Python IDLE's code editor has features like syntax highlighting and code completion that make it simpler and faster to write Python programmes.
•Python IDLE has a built-in debugger that enables programmers to walk through their code and find faults and problems.
•Python IDLE may be used on Linux, macOS, and Windows thanks to its cross-platform nature.
•Python IDLE is included with the Python installation, thus users don't need to install any more programmes in order to begin coding in Python.
•Python IDLE is open-source, free software, which entitles users to use it with no any limitations for both business and non-commercial uses.

How to execute a program

- Open IDLE
- Open new file
- Write down the program
- Save the file with filename with extension as .py Run the program from IDLE or press F5

Python Comments

- Comments are used by the programmer to explain the piece of code to others as well as himself in a simple language.
- As like other programming language python use some special character for commenting
- Comments are preceded by hash symbol(#) on a line.
- Our code is more comprehensible when we use comments in it. It assists us in recalling why specific sections of code were created by making the program more understandable.

Single-Line Comments

Single-line remarks in Python have shown to be effective for providing quick descriptions for parameters, function definitions, and expressions. A single-line comment of Python is the one that has a hashtag # at the beginning of it and continues until the finish of the line. If the comment continues to the next line, add a hashtag to the subsequent line and resume the conversation. Consider the accompanying code snippet, which shows how to use a single line comment:

Code

```
# This code is to show an example of a single-line comment
print( 'This statement does not have a hashtag before it' )
```

Multi-Line Comments

Python does not provide the facility for multi-line comments. However, there are indeed many ways to create multi-line comments.

With Multiple Hashtags (#)

In Python, we may use hashtags (#) multiple times to construct multiple lines of comments. Every line with a (#) before it will be regarded as a single-line comment.

Code

```
# it is a
# comment
# extending to multiple lines
```

Docstring Comments

Python docstrings are the string literals that appear right after the definition of a function, method, class, or module.

```
def square(n):
    '''Takes in a number n, returns the square of n'''
    return n**2

square(5)
25
```

Python __doc__ attribute

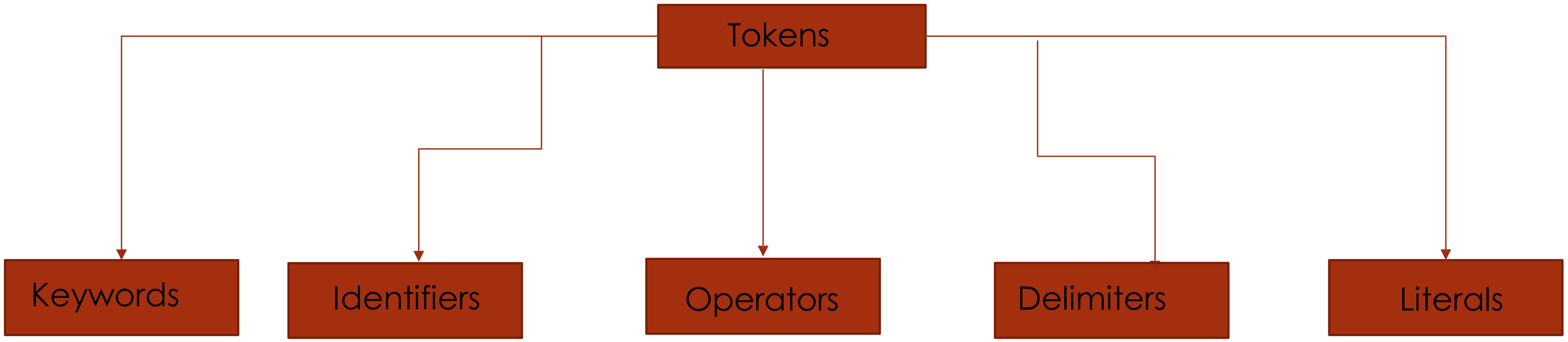
Python Whenever string literals are present just after the definition of a function , module, class, or method ,they are associated with the object as their __doc__ attribute. We can later use this attribute to return the docstring.

```
def square(n):
    '''Takes in a number n, returns the square of n'''
    return n**2

print( square. __doc__)
Takes in a number n, returns the square of n
```

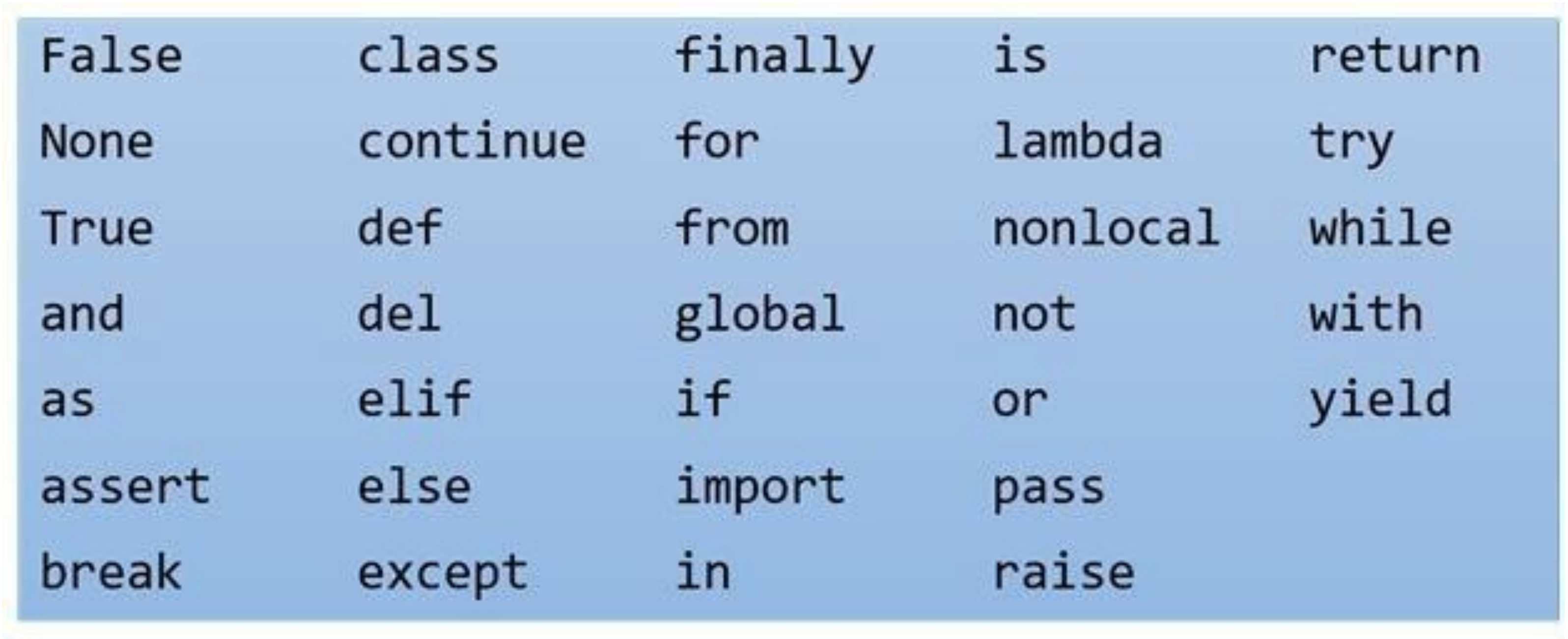
Token

- A program in python contains a sequence of instructions. Python breaks each statement into a sequence of lexical components known as tokens.
- Each token corresponds to a substring of a statement.
- Python contains various types of token



Keywords

- These are the reserved words with fixed meanings assigned to them.
- Keywords cannot be used as identifiers or variables.



False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Identifier / Variable

- This is the name used to find the variable, function, class or other objects
- All Identifier must obey the following rules
  - Is a sequence of characters that consists of letters, digits and underscore
  - Can be any of length
  - Starts with letter which can be either lower or upper case
  - Can start with underscore '_'.
  - Cannot start with a digit
  - Cannot be a keyword
- Eg valid Identifier:Name,Roll_No, A1_Address
- Eg for invalid Identifiers : for,Salary@,12Name,First Name

Delimiter

- These are the symbols that perform a special role in Python like grouping, Punctuation and assignment
- Python use the following symbols and symbol combinations as delimeters
  - ( ), [ ] , { }
  - , : , . , ' ,= , ;
  - += -= *= /+ //= %=
  - &= |= ^= >>= <<= **=

Literals

- These are the numbers or strings or characters that appear directly in a program.
- Types of literals in python
- Integer Literal
- Floating point Literal
- Character Literal
- String Literal
- Python also uses other types of literals like lists, tuple and dictionary

Data Types

- The data stored in the memory can be of many types.

  Eg :Name in Alphabetic, Address in Alphanumeric
- Python has six basic data types
  - Numeric
  - String
  - List
  - Tuple
  - Dictionary
  - Boolean
  - Sets

Numeric

- Numeric data can be broadly classified into 2.
  - Integers and Real Numbers.
  - Octal , hexadecimal and complex numbers also comes under numeric datatype.
- Integers are the combination of positive and negative numbers including zero
- The floating point numbers contain decimal and a fractional part.
- Eg :
  - >>10
  - Num=12222120
  - C=-87
  - 2.5

String

- Single ,double or Triple quotes are used to represent strings.
- A String in python can be a series or sequence of alphabets, numerals and special characters.Strings are arrays of characters.
- Like C the first character of string has an index 0
- There are many operations that can be performed in strings such as
  - Slice operator ([ ] , :). (a='hello' print(a[4],print(a[0:3]))
  - Concatenation operator (+) (a='hello', b='world' print(a+b))
  - Repetition Operator (*). Print(a*3)

Len() to find length of string

Looping through string . eg- for a in "banana":

```
                            print(a)
```

check string. Eg x= "I am , ok"

```
                 print("ok" in x)
```

returns true .

x.upper(),x.lower(),x.strip(),x.replace("a","z"),x.split(",")

String

- One of Python's coolest features is the string format operator %. This operator is unique to strings and makes up for the pack of having functions from C's printf() family
- Eg
```
print "My name is %s and weight is %d kg!" % ('Zara', 21)
```
```
Out put    : My name is Zara and weight is 21 kg!
```
- Raw strings do not treat the backslash as a special character at all. Every character you put into a raw string stays the way we wrote it
  Eg : print ("hello \newline") shows   hello \newline
- print() is the function used to print string literals.

List

- A list can contain the same type of items and different type of items
- It is an **ordered** and **indexable** sequence.
- To declare a list we need to separate the items using commas and enclose them within **square brackets**([ ]).
- Similar like String the list also have (+), (*),(:) operators for concatenation, repetition and sublists respectively
- List items are ordered, **changeable**(mutable), and allow duplicate values.
- a=[1,’abc’,67]

  a[1]=‘efg’
- a.insert(2,’xxx’) (a.append(),a.extend(b),a.remove(),a.pop(1),a.clear())

  print(a)

Tuple

- Similar to list tuple is used to store sequence of items(tuple is **ordered** and **duplicates allowed**).
- Like list Tuple consists of items separated by commas but enclosed within **parenthesis**.
- The main differences between Tuple and Lists
  - In list terms are enclosed in square brackets but in tuples it is enclosed by parenthesis
  - Lists are mutable(changeable) where as Tuples are immutable(**unchangeable**). Tuples are **read only**.

  Len(),append(),remove(),count(),index(),+,*,:

Dictionary

- It is an **unordered**(version 3.6) ,changeable collection of **key-value** pairs.
- Keys and values can be of **any type** in dictionary
- Items in dictionary are enclosed in **curly-braces**{} and separated by comma(,).No duplicates are allowed.
- A column (:) is used to separate key and values
- Eg
  - dict1={1:'first',2:'Second'}

  Print(dict1[2]) or dict1.get(2)

  Len(),dict1.keys(),dict1.values(),dict1.update({2:"two"}).dict1.copy(),

  clear(),pop()

Boolean

- The True or False data is known as Boolean Data and the data types which stores this Boolean data are known as Boolean Data Types.
- Eg
  - X=5

    Y=10

    Print(bool(x==y)) will be false.

    Print(bool(x<y)) will be true.

  Boolean operators are AND,OR,NOT,EQUAL,and NOT-EQUAL.

Sets

- Sets are the one kind of data type which have **unordered** collection of data.
- A set does **not contain any duplicate values** or elements.
- Operations that can be performed on two sets
  - **Union**: It returns all the elements from both the sets. It is performed by using & operator
  - **Intersection** :It returns all the element which are common or both in the sets. It is performed by | operator
  - **Symmetric Difference**: It returns the element which are present in either set1 or set2 but not in both. It is performed by using ^ operator

Sets

- **Set Items**

  Set items are unordered, unchangeable, and do not allow duplicate values

- **Unordered**

  Unordered means that the items in a set do not have a defined order.

  Set items can appear in a different order every time you use them, and cannot be referred to by index or key.

- **Unchangeable**

  Sets are unchangeable, meaning that we cannot change the items after the set has been created.

- **Duplicates Not Allowed**

  Sets cannot have two items with the same value.

Assignment 1.1: Explain different Types of Data types in Python?

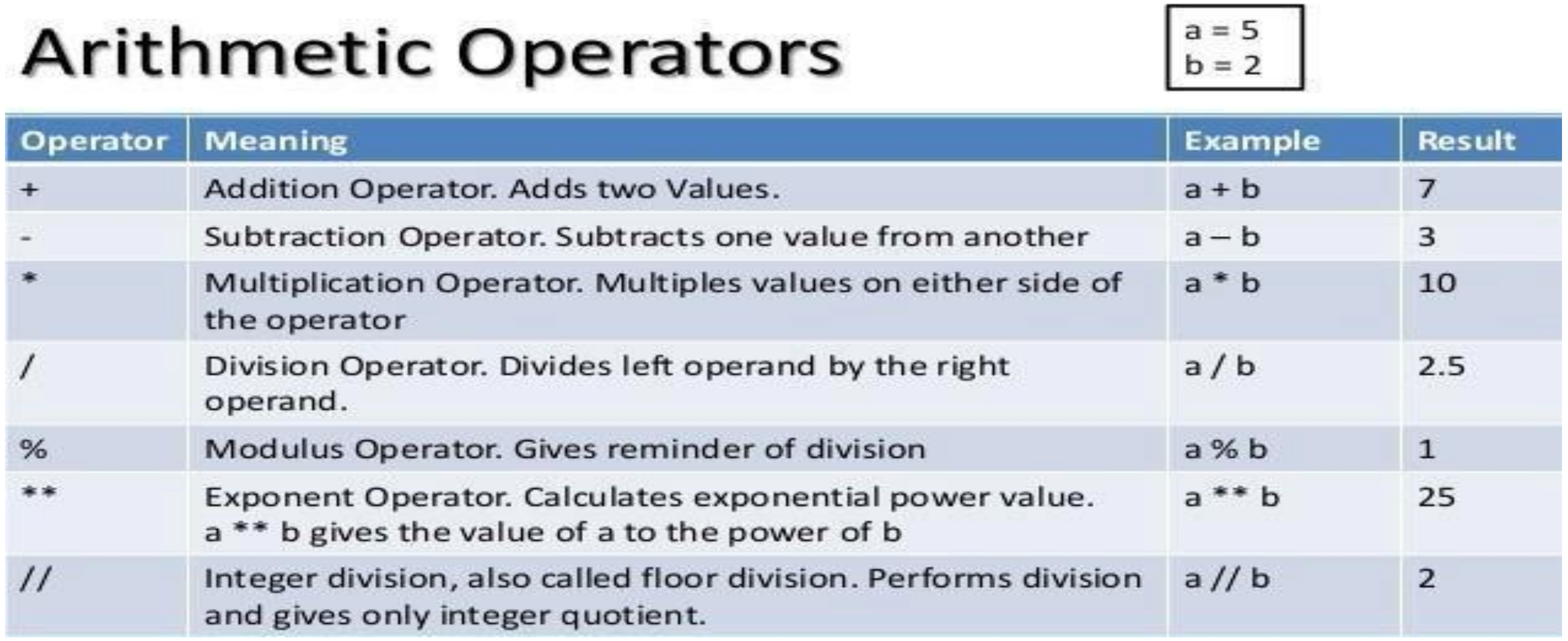
Hint:
Numerals
Strings
Lists
Tuple
Dictionary
Sets
Boolean

Operator

- An Operator indicates an operation to be performed on data to yield a result.
- Based on functionality ,Operators are categorizes into following seven types
  - Arithmetic Operator
  - Comparison Operator
  - Assignment Operator
  - Logical Operator
  - Bitwise Operator
  - Membership Operator
  - Identity Operator

Arithmetic Operator

- It mainly classifies into two
  - Unary and Binary
- Unary Operand perform mathematical operation on one operand only(Eg: -,+)
- Binary Operators requires two operands



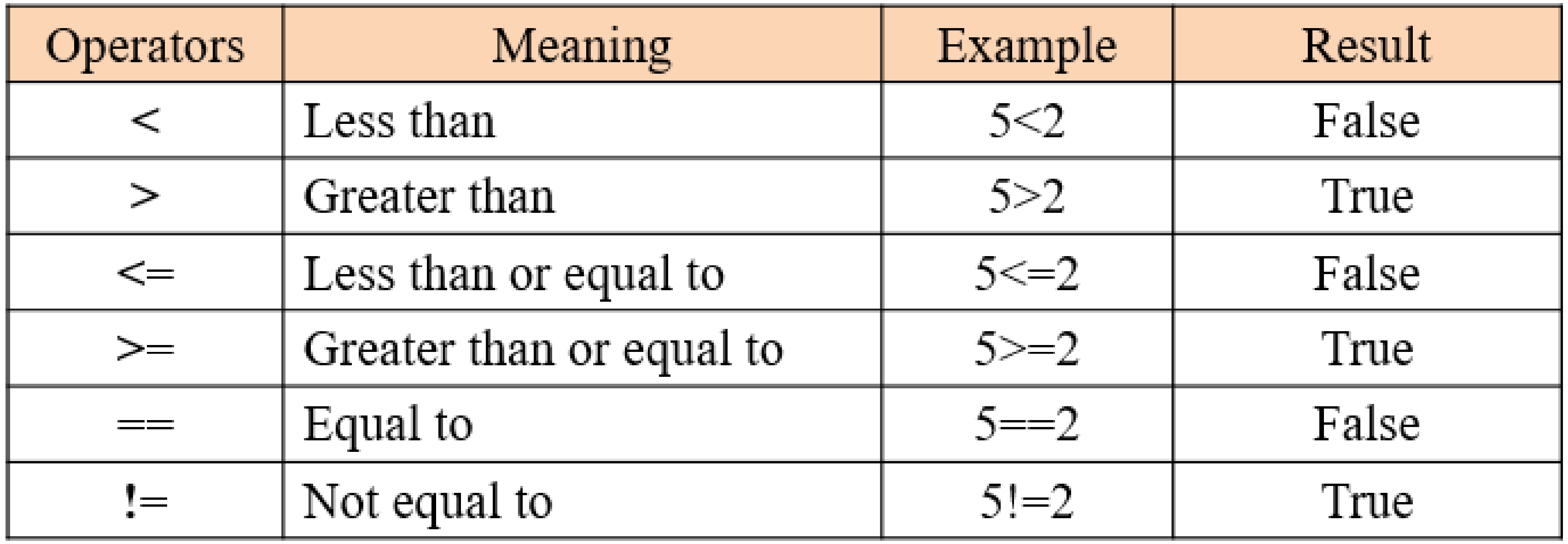
Arithmetic Operators

a = 5
b = 2

Operator	Meaning	Example	Result
+	Addition Operator. Adds two Values.	a + b	7
-	Subtraction Operator. Subtracts one value from another	a – b	3
*	Multiplication Operator. Multiples values on either side of the operator	a * b	10
/	Division Operator. Divides left operand by the right operand.	a / b	2.5
%	Modulus Operator. Gives reminder of division	a % b	1
**	Exponent Operator. Calculates exponential power value. a ** b gives the value of a to the power of b	a ** b	25
//	Integer division, also called floor division. Performs division and gives only integer quotient.	a // b	2

Comparison Operator

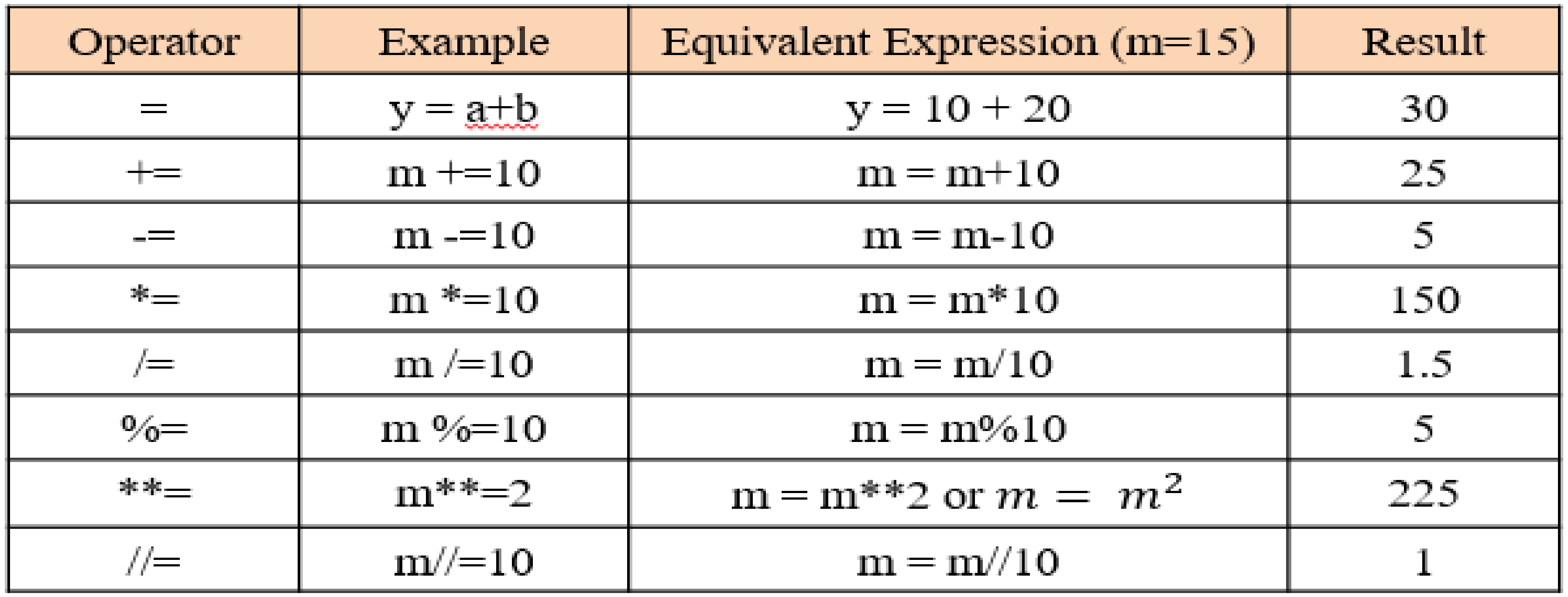
- These operators are used to compare values. These are also called relational operators. The result of these operators is always a Boolean value



Operators	Meaning	Example	Result
<	Less than	5<2	False
>	Greater than	5>2	True
<=	Less than or equal to	5<=2	False
>=	Greater than or equal to	5>=2	True
==	Equal to	5==2	False
!=	Not equal to	5!=2	True

Assignment Operators

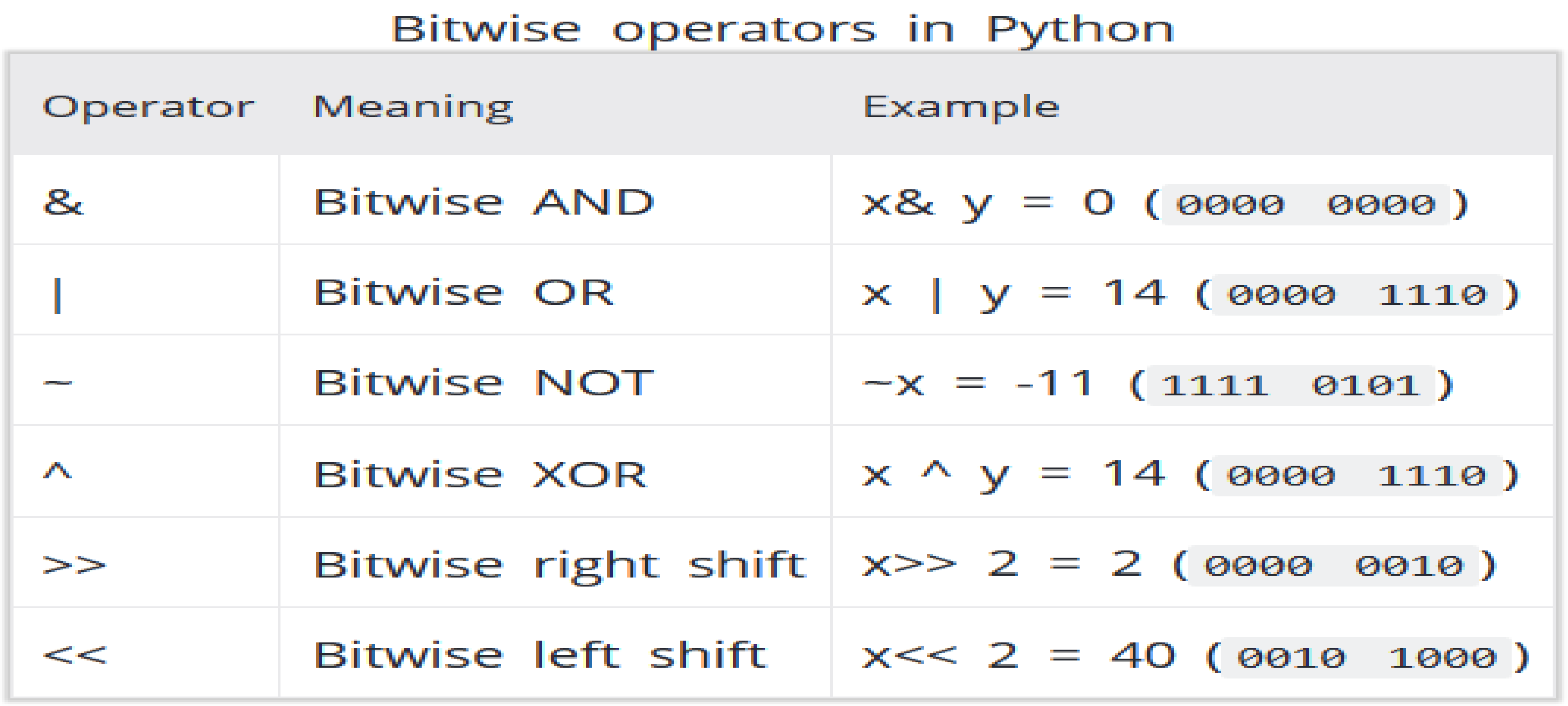
- These operators is used to store right side operand in the left side operand.



Operator	Example	Equivalent Expression (m=15)	Result
=	y = a+b	y = 10 + 20	30
+=	m +=10	m = m+10	25
-=	m -=10	m = m-10	5
*=	m *=10	m = m*10	150
/=	m /=10	m = m/10	1.5
%=	m %=10	m = m%10	5
**=	m**=2	m = m**2 or $m = m^2$	225
//=	m//=10	m = m//10	1

Bitwise Operators

- These operators perform bit level operations on operands.

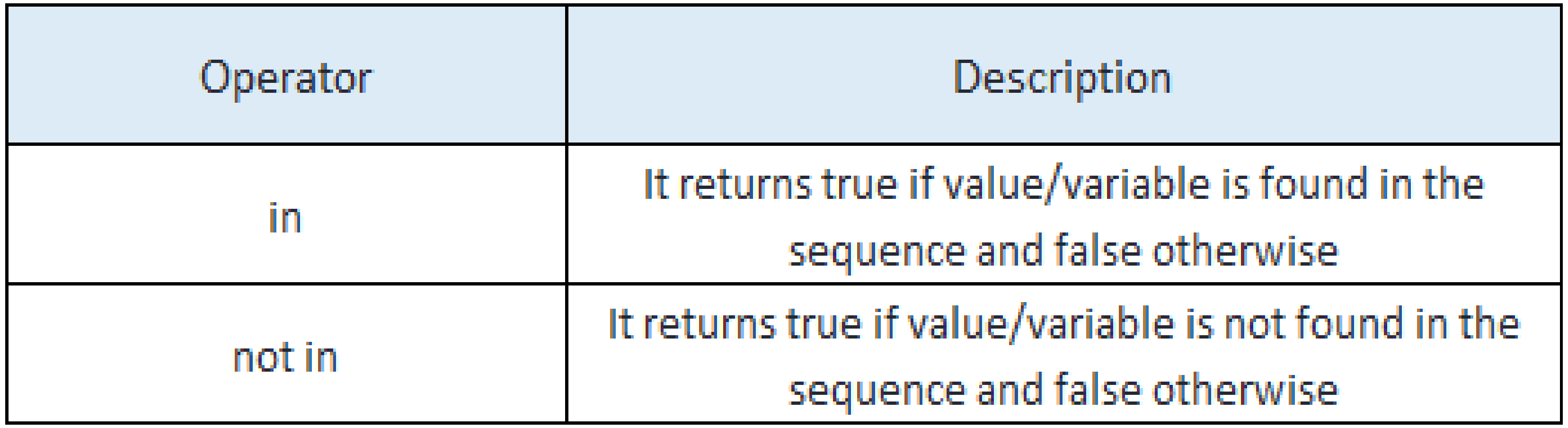


Bitwise operators in Python

Operator	Meaning	Example
&	Bitwise AND	x& y = 0 (`0000 0000`)
|	Bitwise OR	x | y = 14 (`0000 1110`)
~	Bitwise NOT	~x = -11 (`1111 0101`)
^	Bitwise XOR	x ^ y = 14 (`0000 1110`)
>>	Bitwise right shift	x>> 2 = 2 (`0000 0010`)
<<	Bitwise left shift	x<< 2 = 40 (`0010 1000`)

Membership Operators

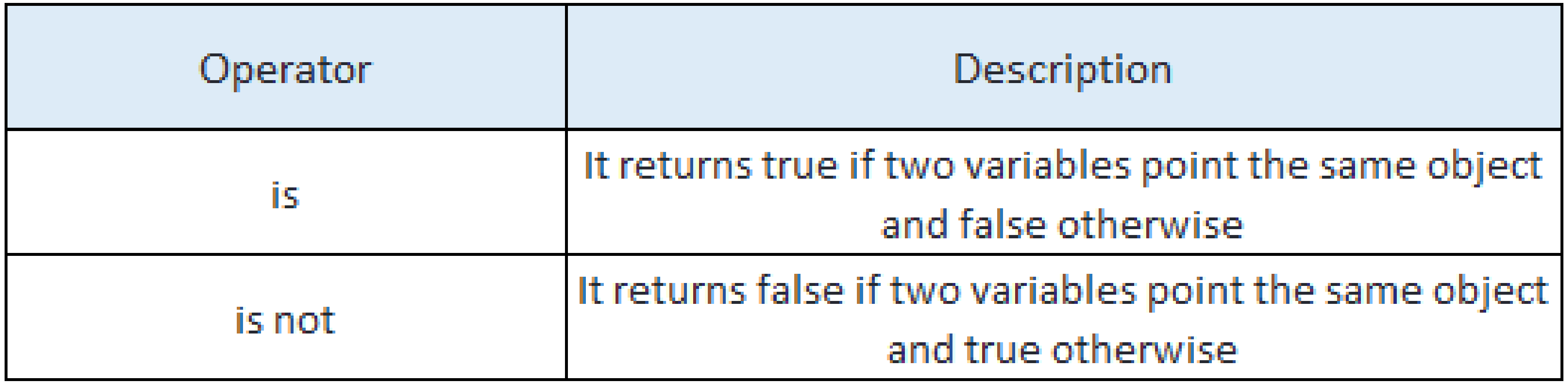
- These operators are used to check an item or an element that is part of a string, a list or a tuple



Operator	Description
in	It returns true if value/variable is found in the sequence and false otherwise
not in	It returns true if value/variable is not found in the sequence and false otherwise

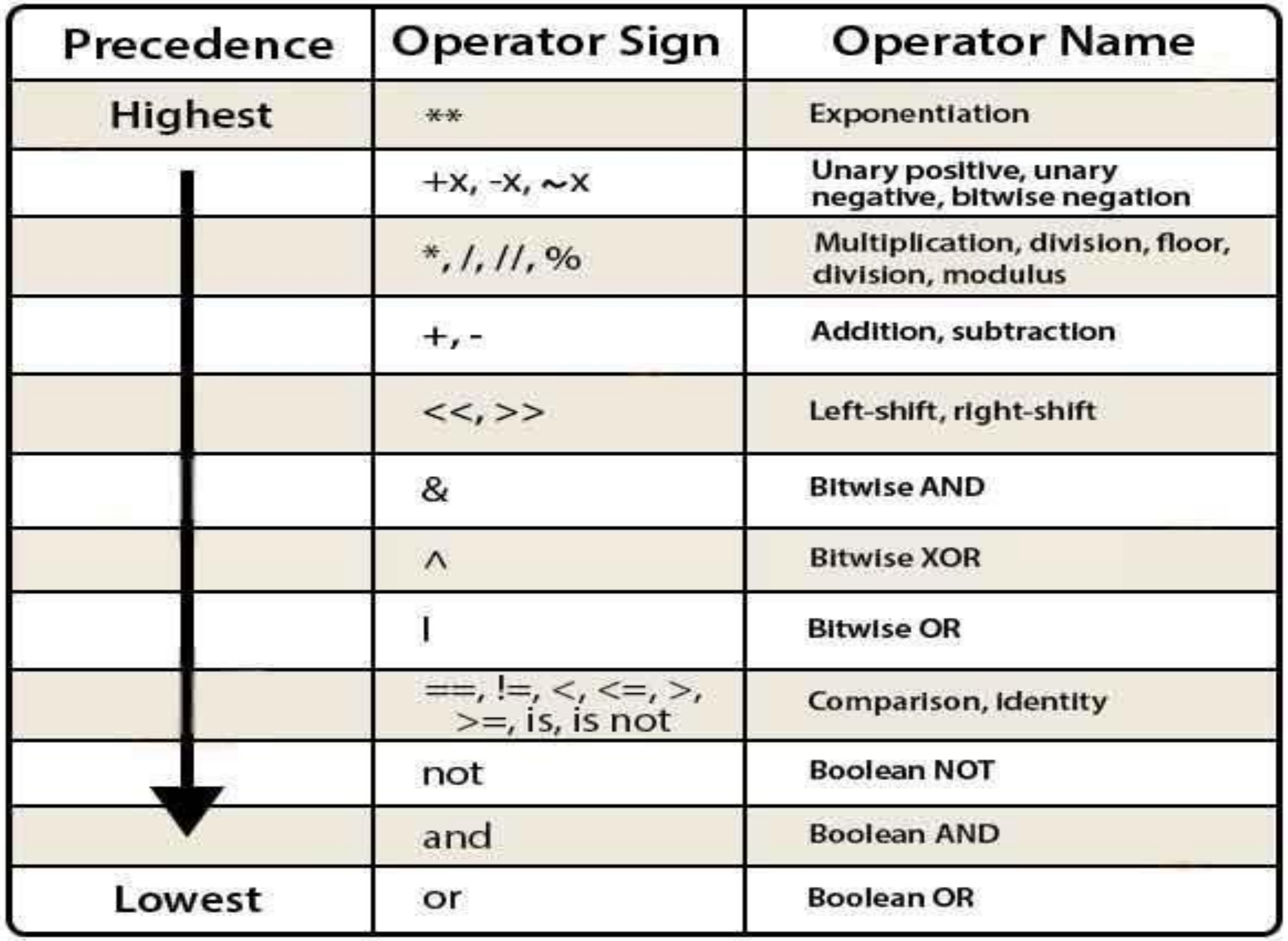
Identity Operator

- These operators are used to check whether both operators are same or not.



Operator	Description
is	It returns true if two variables point the same object and false otherwise
is not	It returns false if two variables point the same object and true otherwise

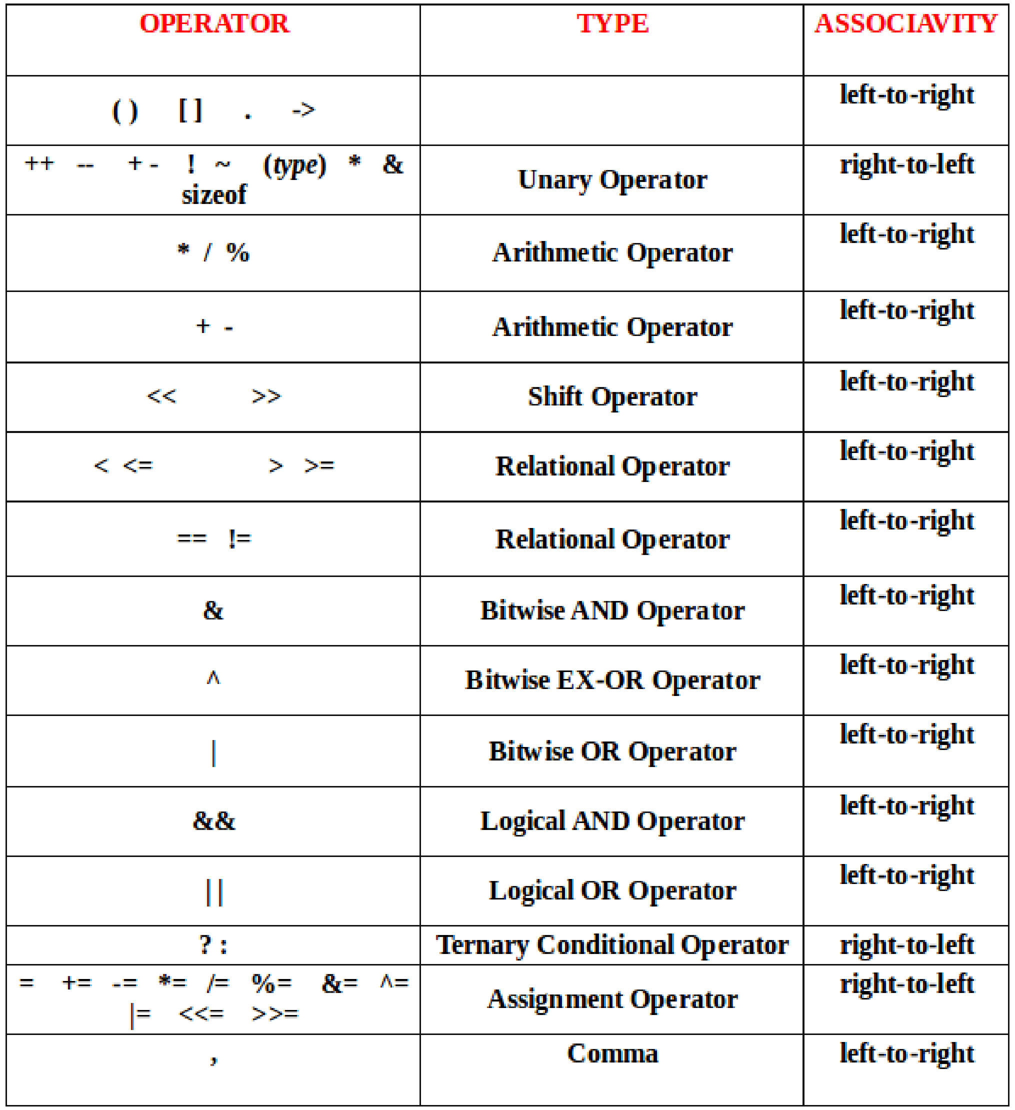
Operator Precedence



Precedence	Operator Sign	Operator Name
Highest	**	Exponentiation
	+x, -x, ~x	Unary positive, unary negative, bitwise negation
	*, /, //, %	Multiplication, division, floor, division, modulus
	+, -	Addition, subtraction
	<<, >>	Left-shift, right-shift
	&	Bitwise AND
	^	Bitwise XOR
	|	Bitwise OR
	==, !=, <, <=, >, >=, is, is not	Comparison, identity
	not	Boolean NOT
	and	Boolean AND
Lowest	or	Boolean OR

Associativity

if an expression contains two or more operators with the same precedence then Operator Associativity is used. It can either be Left to Right or Right to Left.



OPERATOR	TYPE	ASSOCIAVITY
( ) [ ] . ->		left-to-right
++ -- + - ! ~ (*type*) * & sizeof	Unary Operator	right-to-left
* / %	Arithmetic Operator	left-to-right
+ -	Arithmetic Operator	left-to-right
<< >>	Shift Operator	left-to-right
< <= > >=	Relational Operator	left-to-right
== !=	Relational Operator	left-to-right
&	Bitwise AND Operator	left-to-right
^	Bitwise EX-OR Operator	left-to-right
|	Bitwise OR Operator	left-to-right
&&	Logical AND Operator	left-to-right
||	Logical OR Operator	left-to-right
? :	Ternary Conditional Operator	right-to-left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment Operator	right-to-left
,	Comma	left-to-right

Assignment 1.2 : What do you mean by Operator Precedence and Associativity?

Statement and Expression

- **Statement** can be thought as an instruction that can be interpreted by the Python interpreter
- A statement is interpreted by the interpreter and after execution displays some results(If there is a need of displaying it)
- A program can contain statements in sequence. If there are multiple statements the result is displayed after every statement.
- **Expression**: An expression is a combination of variables, operators, values and reserve keyword.
- When ever we type an expression in the command line, the interpreter evaluates it and produces the result

Type functions

- This is a built in method to know the exact type of an value .
- The syntax to know the type of any value is type(value)
- Type function Can take anything as an argument and returns its datatype such as integers, strings,lists,tuple,classes,modules.
- Eg
  - Type(‘Hello world’)

    >>>class ‘str’>

Print function

- The `print()` function prints the specified message to the screen, or other standard output device.

- The message can be a string, or any other object, the object will be converted into a string before written to the screen

- ➨print("Hello how are you?")
- ➨Print(“Hello”,”How are you”sep=“---”)

Eval Function

The `eval()` function evaluates the specified expression, if the expression is a legal Python statement, it will be executed.

Syntax
eval(expressions)

Eg:
```
    x=5
    eval(x+10)
 will return 15.
```

Eg:
```
    expression='x*(x+1)*(x+2)'
    print(expression)
    x=3
    result=eval(expression)
    print(result)
```

Assignment 1.3:Write a program which prompts a user to enter his personal data and print the values along with it types?

- Fields:

  Name

  Age

  College

  Place

  Pincode