OPERATORS

operators

- Operators are the special symbols which are used to perform some specific operation.:-
- To perform any operation it is required to use operator & operands
- In python operators are classified into 7 types:-

1. Arithmetic
2. Relational
3. Assignment
4. Logical
5. Bitwise
6. Membership
7. Identity

1. *Arithmetic*

I. addition (+)
II. subtraction (-)
III. multiplication (*)
IV. division ----->a)normal division / b) floor division // c) modules %
V. power (**)

I. addition (+):-

- Addition is on operator which is used to find sum between two or more operands.
- Syntax :- operand1 + operand 2 (addition # for single value data type)
  collection1 + collection2 (concatenation for collection data types)
- we can add two single value operands of any type but in case of concatenation both the collection should be of same type.

- Concatenation will not support for set and dictionary.
- Examples :-

```
>>>10 + 23
>>> 33
>>> 12 +3.4
>>>  15.4
>>> 12+(7+1j)
>>> (19+1j)
>>> 12 + True
>>> 13
>>> 2.3+4.5
>>> 6.8
>>> 1.2 + (7+3j)
>>> (8.2+3j)
>>> 4.5 + False
>>> 4.5
```

```
>>>  “hello” + “hai”
>>> “hellohai”

>>> [1,2,3] + [5,6,7]
>>> [1,2,3,5,6,7]

>>> (2,3,4,5) + (8,9)
>>> (2,3,4,5,8,9)

>>> {2,4,6} + {3,5,7}
>>> error

>>> ‘hello’ + 3
>>> error

>>> (4,5) + [8,9]
>>>  error

>>> {‘a’:1,’b’:2} + {‘a’:5,’d’:6}
>>> error
```

II) Subtraction :-

>>> subtraction is on operator which is used to find the difference between two or more operands.

>>> syntax `operand1 – operand2`

>>> subtraction operator will support for all the single value DT and only set to set collection

Examples :-

```
>>> 5-2
>>> 3
>>>  3-7.8
>>> -4.8
>>> 2 – (7+4j)
>>> -5-4j
>>> 'hai' – 'hello'
>>> error
>>> 'hello' – [5,9]
>>> error
>>> [9,6,8] – [4,6,8]
>>> error
>>> 4-True
>>> 3
```

```
>>> 3.5-2.1
>>> 1.4

>>> {1,2,3,4} – {1,2,5,6}
>>> {3,4}
>>> # in case of the two sets are compared the values,it will consider only the first & compares it with the second set & gives the values

>>> {3,4,7,8} –{3,8,1,4}
>>> {7}

>>> {1,2,3,4} – {3,6,4}
>>> {1,2}
```

iii) Multiplication (*) :-

>>> multiplication is on operator which is used to find the product b/w two or more operands.

>>> syntax

```
op1 * op2
collection *n
```

single value datatype (svdt)

collection DT where n is only integer

>>> # we can multiple one svdt with another svdt but in case of collection we can multiply it with integer number.

>>> multiplication will not support for set and dictionary.

Example :-

>>> 5*2
>>> 10
>>> 5*3.4
>>> 17.0
>>> 5*(3+4j)
>>> (15+20j)
>>>3.4*2.3
>>> 7.819999
>>> 'hello'*'hello'
>>> error

>>> 'hello' *3
>>> 'hellohellohello'

>>> [1,2]*5
>>> [1,2,1,2,1,2,1,2,1,2]

>>> (7,8)*2
>>>(7,8,7,8)

>>> {5,6,7}*2
>>> error
>>> {'a':3} *2
>>> error
>>'abc' *2.3
>>> error

iv)Division:-

- Division is on operator which is used to find the division output between two or more operands.
- Division operator is further classified into 3 types :-

1.True division (/) :-

>> it is a division operator which is used to get the exact division output along with decimal values ,ex :- 10/3 = 3.33333

>> true is will support all the single value datatype but it won't support for collection data type.

2.Floor division (//):-

>> it is a division operator which is used to get the exact division output ,removing the decimal value ex :- 10//3 = 3

>> floor division will support only integer and float data type.

3.Modulus (%) :-

>> it is a division operator which is used to get the remainder of division output.

>> modules operator will support only integer & float data type. Ex 10%3 = 1

Examples :

```
>>> 3.4/2
>>> 1.7
>>>3.4//2
>>> 1
>>> 'hai'/2
>>> error
```

```
>>>(3+4j)/2
>>> (1.5+2j)
>>>(3+4j)//2
>>> error
```

v) **Power operator (**) :-**

>>> It is on operator which is used to multiple operand for n numbers of time.

>>> syntax `operand**n`

>>> n should be on integer

Example :-

```
>>> 3**2
>>> 9
>>> 7.8**2
>>> 60.83999
>>> True**False
>>> 1
>>> 'hai'**2
>>> error
>>> [2,4,6]**3
>>> error
```