Assignment Operator :-

We can combine assignment operator to assign values to the variable.

Eg1 $x = 10$.

We can combine assignment operator with some other operator to form compound assignment operator.

Eg1 $x += 10$

$\Rightarrow x = x + 10$

(+=, -=, *=, /=, %=, //=, **=, &=, |=, ^=, >>=, <<=)

Eg1
```
x = 10
x += 20
print(x) ==> 30
```

Eg1
```
x = 10
x &= 5
print(x) ==> 0
```

Ternary Operator :-

x = first value if condition else second value

If condition is True then first value will be considered else second value will be considered.

Eg 1 :-

```
a, b = 10, 20
x = 30 if a<b else 40
print(x)  # 30
```

Eg: Read two numbers from the keyboard and print minimum value:-

```
a = int (input ("Enter First Number:"))
b = int (input ("Enter Second Number:"))
min = a if a<b else b
print ("Minimum Value:", min)
```

output:-

```
Enter First Number: 10
Enter Second Number: 30
Minimum value: 10
```

Note:- Nesting of ternary operator is possible

Eg 3:- Program for minimum of 3 numbers:-

```
a = int(input("Enter the First Number:"))
b = int(input("Enter the Second Number:"))
c = int(input("Enter the Third Number:"))
min = a if a<b and a<c else b if b<c else c
print("Minimum Value:", min)
```

Eg 4:- Program for maximum of 3 numbers:-

```
a = int(input("Enter First Number"))
b = int(input("Enter Second Number"))
c = int(input("Enter Third Number"))
max = a if a>b and a>c else b if b>c else c
print("Maximum value:", max)
```

Eg1

```
a = int(input("Enter First Number:"))
b = int(input("Enter Second Number:"))
print("Both numbers are equal" if a==b
else "First Number is less than Second
Number" if a<b else "First Number
Greater than Second Number")
```

Output:-

```
Enter First Number: 10
Enter Second Number: 10
Both numbers are equal.

Enter First Number: 10
Enter Second Number: 20
First Number is Less than Second Number

Enter First Number: 20
Enter Scond Number: 10
First Number Greater than Scond Number.
```

Special Operators:-

Python defines the following 2 special operators.

1. Identity Operators.
2. Membership operators.

1. Identity operators:-

We can use identity operators for address comparison.

2 identity operators are available

1. is
2. is not

x1 is x2 returns True if both x1 and x2 are pointing to the same object.

x1 is not x2 returns True if both x1 and x2 are not pointing to the same object.

Eg:

```
a = 10
b = 10
print(a is b)  True

x = True
y = True
print(x is y)  True
```

Eg:

```
a = "Mango"
b = "Mango"
print(id(a))
print(id(b))
print(a is b)
```

Eg:

```
list1 = ["One", "two", "three"]
list2 = ["One", "two", "three"]
print(id(list1))
print(id(list2))
print(list1 is list2)        False
print(list1 is not list2)    True
print(list1 == list2)        True
```

Note:

- We can use is operator for address comparison where as == operator for content comparison.

(2) Membership Operators:-

We can use Membership operators to check whether the given object present in the given collection. (It may be String, List, Set, Tuple or Dict)

in → Returns True if the given object present in the specified collection.

not in → Returns True if the given object not present in the specified collection.

Eg:-

```
x = "hello learning Python is very easy!!!"
print('h' in x)          True
print('d' in x)          False
print('ol' not in x)     True
print('Python' in x)     True
```

Eg:-

```
list1 = ["sunny", "bunny", "chinny", "pinny"]
print("sunny" in list1)          True
print("tunny" in list1)          False
print("tunny" not in list1)      True
```

(7)

Unary operators

Unary operators act on single operands. Python supports unary minus operator. when an operand is preceded by a minus (-) sign, the unary operator negates its value.

Eg: b = 10
a = -b
Print (a) → -10

Operator Precedence:-

If multiple operators present then which operator will be evaluated first is decided by operator precedence

Eg: print (3 + 10*2) → 23

print((3+10) *2) → 26.

The following list describes operator precedence in Python.

Eg:
```
a = 30
b = 20
c = 10
d = 5
print((a+b) * c/d)      100
print((a+b)*(c/d))      100
print(a+(b*c)/d)        70.0
```

Operator precedence chart

**	Exponentiation
~, +, -	Complement, unary plus (positive), minus (negative
*, /, %, //	Multiply, divide, modulo and floor division.
+, -	Addition and subtraction
>>, <<	Right and left bitwise shift
&	Bitwise 'AND'
^ |	Bitwise [redacted] exclusive 'OR' and regular 'OR'
<=, <, >, >=	Comparison operators.
<>, ==, !=	Equality operators.
=, %=, /=, //=, -=, +=, *=, **=	Assignment operators
is, is not	Identity operators.
[redacted] in, not in	Membership operators.
not, or, and	Logical operators.