Operators

---

1) Arithmetic operators ---> + , - , * , / , // , % , **
2) Relational operators ---> > , < , == , >= , <= , !=
3) Logical operators ---> and , or , not
4) Assignment operators ---> = , += , -= , *= , /= , //= , %= , **=
5) Identity operators ---> is , is not
6) Membership operators ---> in , not in
7) Ternary operator ---> op1 if cond else op2
8) Walrus operator ---> :=
9) Bitwise operators ---> & , | , ^ , ~ , << , >>

Arithmetic operators

---

1) What are the seven different arithmetic operators ? ---> + , - , * , ** , / , // , %
2) What does + operator do ? ---> Addition
3) What does - operator do ? ---> Subtraction
4) What does * operator do ? ---> Multiplication
5) What does / operator do ? ---> Performs division and returns float quotient
6) What does // operator do ? ---> Performs division and returns an integer quotient
7) What does % operator do ? ---> Performs division and returns remainder
8) What does ** operator do ? ---> Returns power
9) Which operator has got highest priority in arithmetic operators ? ---> **
   Which operators have got next higher priority ? ---> * , / , // , %
   Which operators have got lower priority ? ---> + , -

+ operator

---

1) What is the result of 10 + 20 ? ---> 30
   What is the result of '10' + '20' ? ---> '1020'

2) What does non-sequence + non-sequence do ? ---> Addition
   What does sequence + sequence do ? ---> Concatenation
3) Is sequence + non-sequence valid ? ---> No
4) What does [10 , 20] + [30 , 40] do ? ---> Concatenates the two lists to form a new list with four elements
   i.e. [10 , 20 , 30 , 40]
5) Which sequences can not be concatenated with + operator (Total : 3) ? ---> range , set and dictionary
6) Is + operator overloaded ? ---> Yes becoz + operator does both addition and concatenation

Find outputs (Home work)

```
print(10 + 20) # 30
print(10.8 + 20.6) # 31.4
print(3 + 4j + 5 + 6j) # 8 + 10j
print(True + False) # 1 + 0 = 1
```

```
print('Hyder' + 'abad') # Hyderabad
print('Sankar' + 'Dayal' + 'Sarma') # SankarDayalSarma
print('10' + '20') # 1020
print([10 , 20 , 30] + [1 , 2 , 3]) # [10,20,30,1,2,3]
print((25 , 10.8 , 'Hyd') + (3 + 4j , True , None)) # (25 , 10.8 , 'Hyd' , 3 + 4j , True , None)
#print({10 , 20} + {30 , 40}) # Error becoz sets can not be concatenated with + operator
#print({10 : 'Hyd'} + {20 : 'Sec'}) # Error becoz dictionaries can not be concatenated with + operator
#print(range(4) + range(5)) # Error becoz range objects can not be concatenated
#print(10 + '20') # Error due to sequence and non-sequence
#print([10 , 20 , 30] + 5) # Error due to sequence and non-sequence
#print([10 , 20 , 30] + (40 , 50 , 60)) # Error becoz list and tuple can not be concatenated

# Find outputs
print({10 , 20} | {30 , 20}) # {10 , 20 , 30} in any order
print({10 : 'Hyd' , 20 : 'Sec'} | {30 : 'Cyb' , 20 : 'Vja'}) # {10 : 'Hyd' , 20 : 'Vja' , 30 : 'Cyb'}
#print(range(4) | range(5)) # Error becoz range objects can not be concatenated
#print([10 , 20] | [30 , 20]) # Error becoz lists can not be concatenated with | operator
'''
1) How to concatenate sets and dictionaries ? ---> With | operator but not with + operator
2) How to concatenate lists , tuple and strings ? ---> With + operator but not with | operator
3) What about range objects ? ---> They can never be concatenated
'''
```

```
# / operator demo program
print(9.0 / 2) # 4.5
print(9 / 2.0) # 4.5
print(9.0 / 2.0) # 4.5
print(9 / 2) # 4.5
print(10.5 / 2) # 5.25
print(10 / 3) # 3.33
print(10 / 2) # 5.0
'''
What does / operator do ? ---> Peforms division and returns float quotient
'''


# // operator demo program
print(9 // 2) # 4
print(9.0 // 2) # prev integer of 4.5 = 4.0
print(9 // 2.0) # prev integer of 4.5 = 4.0
print(9.0 // 2.0) # prev integer of 4.5 = 4.0
print(10.5 // 2)# prev integer of 5.25 = 5.0
print(10 // 3) # 3
print(10.0 // 3) # prev integer of 3.33 = 3.0
print(8.5 // 3) # prev integer of 2.8 = 2.0
print(18 // 4) # 4
print(-18 // 4) # prev integer of -4.5 = -5
print(-(18 // 4)) # -4
'''
// operator
----------
1) What is the result of integer // integer ? ---> Integer quotient
   What is the result of integer // float ? ---> Float quotient with .0
   What is the result of float // integer ? ---> Float quotient with .0
   What is the result of float // float ? ---> Float quotient with .0
2) What is the result of integer / integer ? ---> Float quotient
   What is the result of integer / float ? ---> Float quotient
   What is the result of float / integer ? ---> Float quotient
   What is the result of float / float ? ---> Float quotient
3) What does / operator do ? ---> Performs division and returns float quotient
    What does // operator do ? ---> Performs division and returns integer quotient
4) When is the result of // operator integer ? ---> When both the operands are integers
   When is the result of // operator float with .0 ? --->When at least one operand is float
'''



# floor() and ceil() functions demo program
import math
```

```
print(math . floor(10.8)) # Previous integer of 10.8 i.e. 10
print(math . ceil(10.8)) # Next integer of 10.8 i.e. 11
print(math . floor(25.0)) # 25 due to .0
print(math . ceil(25.0)) # 25 due to .0
print(math . floor(-3.5)) # -4 due to -ve number
print(math . ceil(-3.5)) # -3
print(math . floor(-9.0)) # -9 due to .0
print(math . ceil(-9.0)) # -9 due to .0
print(math . floor(25.1)) # 25
print(math . ceil(25.1)) # 26
#print(floor(3.5)) # Error becoz there is no floor() function in current module
#print(ceil(3.5)) # Error becoz there is no ceil() function in current module
'''
1) What does floor(x) do ? ---> Returns previous integer of 'x'
2) What does ceil(x) do ? ---> Returns next integer of 'x'
'''


# % operator demo program
print(9 % 5) # 4
print(9.0 % 5) # 4.0
print(9 % 5.0) # 4.0
print(9.0 % 5.0) # 4.0
print(10.5 % 2) # 0.5
print(8.9 % 3) # 2.9

'''
% operator
---------------
1) What does % operator do ? ---> Performs division and returns remainder
2) When is the result of % operator integer ? ---> When both the operands are integers
   When is the result of % operator float ? ---> When at least one operand is float
'''


# Find outputs
#print(7 / 0) # Error due to division by zero
#print(7 // 0) # Error due to division by zero
#print(7 % 0) # Error due to division by zero
'''
What does division by zero do ? ---> Throws error
'''


# ** operator demo program
print(3 ** 4) # 3 ^ 4 = 81
print(10 ** -2) # 10 ^ -2 = 0.01
print(4 ** 3 ** 2) # 4 ^ 3 ^ 2 = 4 ^ 9
```

```
print(3 + 4 * 5 - 32 / 2 ** 3) # 19.0
'''
** operator
----------
1) What is ** operator called ? ---> Exponential operator
2) What is the result of a ** b ? ---> a ^ b
3) What is the result of a ** b ** c ? ---> a ^ b ^ c
4) What is the associativity of ** operator ? ---> Right to left
   What is the associativity of remaining arithmetic operators ? ---> Left to right
5) 3 + 4 * 5 - 32 / 2 ** 3
   = 3 + 4 * 5 - 32 / 8
   = 3 + 20 - 32 / 8
   = 3 + 20 - 4.0
   = 23 - 4.0
   = 19.0
'''
```

Relational Operators

---

1) What are the six different relational operators ? ---> > , < , == , >= , <= , !=

2) What does relational operators do ? ---> Compares objects but not references

   How to compare references ? ---> With is operator

3) What is the result of relational operators ? ---> True (or) False

4) Which operators have got higher priority in relational operators ? ---> > , < , >= , <=

   Which operators have got lower priority ? ---> == , !=

5) Which operators have got higher priority between arithmetic and relational operators ? ---> Arithmetic operators

```
# Relational operators demo program (Home work)
print(9 >= 5) # True becoz > is satisfied
print(9 >= 9) # True becoz = is satisfied
print(9 >= 12) # False becoz both are not satisfied
print(6 <= 8) # True becoz < is satisfied
print(6 <= 6) # True becoz = is satisfied
print(6 <= 4) # False becoz both are not satisfied
print(9 != 7) # True
print(6 == 8) # False
print(True > False) # 1 > 0 is True
print(3 + 4j == 3 + 4j) # True
print(3 + 4j == 5 + 6j) # False
print(3 + 4j != 5 + 6j) # True
print(10 == 10.0) # True becoz 10 and 10.0 are same
#print(3 + 4j > 3 + 4j) # Error becoz complex numbers can not be compared with >
'''
1) Can complex numbers be compared ? ---> Yes with == and !=
2) When can complex numbers be not compared ? ---> Wiith > , < , >= and <=
'''
```

```
# Find outputs (Home work)
print('Rama' > 'Rajesh') # True becoz 'm' > 'j'
print('Rama' < 'Sita') # True becoz 'R' < 'S'
print('Hyd' == 'Hyd') # True due to same strings
print('Rama' <= 'Ramana') # True becoz '' <= 'n'
print('Rama Rao' >= 'Rama') # True becoz ' ' >= ' '
print('Hyd' != 'Sec') # True becoz strings are different
print('HYD' < 'hyd') # True becoz 'H' < 'h'
'''
1) Can strings be compared with > , < , == , >= , <= and != ? ---> Yes only in python but not in other languages
2) What are compared internally when strings are compared ? ---> 1st non-matching characters
3) Are characters compared (or) their unicode values ? ---> Unicode values
```

4) How many unicode values exist ? ---> 512

5) What is the range of unicode values ? ---> 0 to 511

6) What are the unicode values of 'A' to 'Z' ? ---> 65 to 90

What are the unicode values of 'a' to 'z' ? ---> 97 to 122

What are the unicode values of '0' to '9' ? --->48 to 57

What is the unicode value of '$' ? ---> 36

What is the unicode value of space ? --->32

7) What is another name of unicode ? ---> Extended Ascii (American standard code for information and interchange)

'''

Chaining relational opeartors (Home work)

print(10 < 20 < 30) # True becoz both are satisfied

print(10 >= 20 < 30) # False becoz 1st cond is not satisfied

print(10 < 20 > 30) # False becoz 20 is not > 30

print(1 < 2 < 3 < 4) # True becoz all are satisfied

print(1 < 2 > 3 > 1) # False becoz 2 is not > 3

print(4 > 3 >= 3 > 2) # True becoz all are satisfied

'''

1) Can relational operators be chained ? ---> Yes only in python but not in other languages

2) When is the result True ? ---> When all the conditions are True

When is the result False ? ---> When at least one condition is False

'''

Logical operators

---

1) What are the three different logical operators ? ---> and , or , not

2) Which operator has got highest priority in logical operators ? ---> not

Which operator has got 2nd higher priority ? ---> and

Which operator has got lowest priority ? ---> or

3) Which operator has got higher priority between relational and logical operators ? ---> Relational operators

and operator demo program

print(True and False) # False

print(False and True) # False

print(False and False) # False

print(True and True) # True

print(10 and 20) # 20

print(0 and 20) # 0

print(-25 and 0) # 0

print('' and 25) # Empty string

```
print(6j and 'Hyd') # Hyd
print(0j and 'Sec') # 0j
print('Hyd' and 10.8) # 10.8
print(10 and 20 and 30) # 20 and 30 = 30
'''
and operator
----------
1) When is the result of and operator False ? ---> When at least one operand is False
   When is the result of and operator True ? --->When both the operands are True
2) What is the result of op1 and op2 when op1 is True ? ---> op2
   What is the result of op1 and op2 when op1 is False ? ---> op1
3) Is 0 True (or) False ? ---> False
   What about Non-zero ? ---> True
4) Is '' True (or) False ? ---> False due to empty string
    What about non-empty string ? ---> True
5) What is 0j ---> False due to zero imag
   What is 4j ---> True due to non-zero imag
   What is 3 + 0j ? ---> True due to non-zero real
'''



# or operator demo program
print(True or False) # True
print(False or True) # True
print(True or True) # True
print(False or False) # False
print(10 or 20) # 10
print(0 or 20) # 20
print(-25 or 0) # -25
print('' or 35) # 35
print(6j or 'Hyd') # 6j
print(0.0 or 3 + 4j) # 3+4j
print('Hyd' or 10.8) # Hyd
'''
or operator
----------
1) When is the result of or operator True ? ---> When at least one operand is True
   When is the result of or operator False ? ---> When both the operands are False
2) What is the result of op1 or op2 when op1 is False ? ---> op2
   What is the result of op1 or op2 when op1 is True ? ---> op1
3) and , or operators are quite opposite
'''



# not operator demo program
```

```
print(not True) # False
print(not False) # True
print(not 25) # not True is False
print(not 0) # not False is True
print(not 'Hyd') # not True is False
print(not '') # not False is True
print(not -10) # not True is False
print(not not 'Hyd') # not not True= not False = True
'''
not operator
----------
1) What does not operator do ? ---> Complement operation
2) Is not a unary operator ? ---> Yes due to single operand
   What about and , or ? ---> Binary operators due to two operands
3) What is the associativity of unary operators ? ---> Right to Left
   What is the associativity of binary operators ? ---> Left to Right except for **
'''



# Find outputs (Home work)
i = 10
i = not i > 14 # i =not false ---> i = True
print(i) # True
print(not(6 < 4 or 9 >= 5 and 6 != 6)) # not(False or True and False) = not(False or False) = not False = True
'''
1) i = not i > 14
  Which operator is first evaluated and why ? --->
        > becoz relational operator '>' has got higher priority over logical operator not
2) not(6 < 4 or 9 >= 5 and 6 != 6)
   = not(False or True and False)
   = not(False or False)
   = not False
   = True
'''
```

Assignment operators

---

1) =
2) +=
3) -=
4) *=
5) /=
6) //=
7) %=
8) **=

Assignment Statement

---

1) What is = operator called ? ---> Assignment operator
2) What does = operator do ? ---> Assigns reference to an object
3) What is the statement with = operator called ? ---> Assignment statement
4) What does a = 25 do ? ---> Assigns reference to object 25
5) a = [10 , 20 , 15 , 18]
   b = a
   What does b = a do ? ---> Assigns reference 'b' to same list where 'a' points
   Finally both the references point to same list
6) a = 4
   b = 5
   What does c = a + b * 6 do ? ---> Assigns reference 'c' to object 34(Result of the expression)

```
# abs() function demo program
from builtins import abs # Optional becoz abs is automaticallyy imported
print(abs(-35.8)) # 35.8
print(abs(-27)) # 27
print(abs(29.5)) # 29.5
print(abs(32)) # 32
import builtins # Mandatory becoz builtins module is not imported automcatically
print(builtins . abs(-25)) # 25
'''
```

1) What is the simiarity between abs() and fabs() ? ---> Both the functions convert -ve value to +ve value
2) What is the result of abs(integer) ? ---> Positive integer
   What is the result of abs(float) ? ---> Positive float
   What is the result of fabs(integer (or) float) ? ---> Always float
3) Can abs() function be called without import ? ---> Yes becoz it is a function of builtins module
   and hence automatically imported
   Can fabs() function be called without import ? --->
   No becoz it is not automatically imported as it is a function of math module

'''

# Assignment operators demo program (Home work)
a = 25
print(a) # 25
b = a
print(b) # 25
print(a is b) # True
x = 4
y = 5
z = x + y * 6
print(z) # 34
#25 = a # Error becoz 25 is not a reference
#a + b = x + y # Error becoz a + b is not a reference
'''
1) a = 25
   What does b = a do ? ---> Assigns reference 'b' to the same object where 'a' points
                i.e. Reference copy but not object copy
2) In other words b = a does not copy value of object 'a' to object 'b'
3) id is copied but not value
4) a = 25
   b = a
   Why is 25 not copied from object 'a' to 'b' ? --->
            Since there can not be multiple int objects with same value 25
'''

# max() and min() functions demo program
from builtins import max , min # Optional becoz they are automatically imported
print(max(10.8 , 20.6)) # 20.6
print(min(10.8 , 20.6 , 5.9 , 12.3)) # 5.9
print(max(25 , 10.8)) # 25
import builtins # Mandatory becoz module is not imported automatically
print(builtins . max(10 , 20 , 30)) # 30
print(builtins . min(10 , 20 , 15 , 5 , 12)) # 5
'''
How many arguments can max() and min() functions take ? --->
             Any number of arguments becoz they are var-arg functions
'''

# pow() function demo program
from builtins import pow
print(pow(10 , -2)) # 10 ^ -2 = 0.01
print(pow(4 , pow(3 , 2))) # 4 ^ 3 ^ 2 = 4 ^ 9

```
import builtins
print(builtins . pow(2 , 3)) # 2 ^ 3
print(builtins . pow(-2 , -3)) # -2 ^ -3
'''
1) Where is pow() function defined ? ---> In builtins module and also in math module
2) What are the four ways to obtain a ^ b ? ---> a ** b , math . pow(a , b) , builtins . pow(a , b) and pow(a , b)
'''


= Vs ==
---------------
1) What is = operator called ? ---> Assignment operator
   What is == operator called ? ---> Relational operator
2) What does = operator do ? ---> Assigns reference to an object
    What does == operator do ? ---> Compares objects
3) What does a = 7 do ? ---> Assigns reference 'a' to int object '7'
    What does a == 7 do ? ---> Compares value of object 'a' with 7
4) What is the result of the statement a = 7 ? ---> No result
    What is the result of a == 7 ? ---> True / False
5) Are a = b and b = a same ? ---> No and they are quite opposite
   Are a == b and b == a same ? ---> Yes and objects 'a' and 'b' are compared
6) What is operand1 in operand1 = operand2 ? ---> Only reference
   What is operand1 in operand1 == operand2 ? ---> Anything
            i.e. Either reference (or) object
   What is operand2 for both = and == ? ---> Anything



# Find outputs (Home work)
a = b = c = 25 # References a b and c point to same object 25
print(id(a)) # Address of object 25 (may be 1000)
print(id(b)) # Same address
print(id(c)) # Same address
print(a , b , c) # 25 <space> 25 <space> 25
'''
a = b = c = 25
How many objects are there ? ---> Just one object with three references
'''



# Multiple Assignment (Home work)
x , y , z = 25 , 10.8 , 'Hyd' # Multiple assignment
print(x) # 25
print(y) # 10.8
print(z) # Hyd
'''
x , y , z = 25 , 10.8 , 'Hyd'
```

How to divide the above statement into three statements ? ---> x = 25
y = 10.8
z = 'Hyd'
'''

# Find outputs
a = 7
print(a) # 7
a += 5 # a = a + (5) ---> a = 7 + (5) = 12
print(a) # 12
'''
1) What is the expansion of LHS op= RHS ? ---> LHS = LHS op (RHS) where op is any operator
2) What does a += 5 do ? ---> Adds 5 to value of object 'a'
'''
Find outputs (Home work)
a , b , c = 3 , 4 , 5
a *= b + c # a = a * (b + c) ---> a = 3 * 9 = 27
print(a) # 27

# Find outputs (Home work)
a = 20
a %= 3 + 2 * 4 # a = a % (3 + 2 * 4) ---> a = 20 % 11 = 9
print(a) # 9.

# Find outputs (Home work)
a = 3
a **= 4 # a = a ** (4) ---> a = 81
print(a) # 81

```
# Identity operators demo program
a = 25
b = 25
print(a is b) # True
print(a is not b) # False
print(a == b) # True
'''
Identity operators
-----------------------
1) What are the two identity operators ? ---> is and is not
2) What does ref1 is ref2 do ? ---> Compares references
    What does ref1 == ref2 do ? ---> Compares objects pointed by ref1 and ref2
3) What is the result of ref1 is ref2 ? --->
            True when both the references point to same object and False otherwise
   What is the result of ref1 == ref2 ? --->
                                        True when both the objects have same value and False otherwise
4) is and is not are quite opposite operators
'''



# Find outputs (Home work)
a = 25
b = 25.0
print(a is b) # False
print(a is not b) # True
print(a == b) # True becoz 25 and 25.0 are same
'''
Are 25 and 25.0 same ? ---> Yes
'''



# Find outputs (Home work)
a = 'Hyd'
b = 'Hyd'
print(a is b) # True
print(a is not b) # False
print(a == b) # True due to same strins
print()
x = [1 , 2 , 3 , 4]
y = [1 , 2 , 3 , 4]
print(x is y) # False
print(x is not y) # True
print(x == y) # True
print()
```

```
m = (1 , 2 , 3 , 4)
n = (1 , 2 , 3 , 4)
print(m is n) # True
print(m is not n) # False
print(m == n) # True
print(x == m) # False becoz x and m point to different objects
'''
list == tuple
What does == do ? ---> Compares references but not objects becoz they are different class objects
'''



# Membership operators demo program (Home work)
list = [10 , 20 , 15 , 12 , 18]
print(15 in list) # True
print(19 in list) # False
print(14 not in list) # True
print(15 not in list) # False
s = 'Hyd is green city'
print( 'is' in s) #True
print('was' in s) # False
print('g' in s) # True
print('z' in s)# False
print(' ' in s) # True
print('gre' in s) # True
print('yd i' in s) # True
print('' in s) # True due to empty string
print('' not in s) # False
'''
Membership operators
-----------------------------
1) What are the two membership operators ? ---> in and not in
2) What is the syntax of 'in' operator ? ---> element in sequence
3) What does in operator do ? ---> Returns True when element is in the sequence and False otherwise
4) What does not in operator do ? ---> Quite opposite to in operator
'''



Precedence of operators
-----------------------------
1) Which operator has got highest priority ? ---> ()
2) Which operator have got 2nd highest priority ? ---> Arithmetic operators
3) Which operators have got 3rd highest priority ? ---> Relational operators
4) Which operators have got 4th highest priority ? ---> Assignment operators
```

i.e. = , += , -= , *= , /= , //= , %= , **=

5) Which operators have got 5th highest priority ? ---> Identity operators

i.e. is , is not

6) Which operators have got 6th highest priority ? ---> Membership operators i.e. in , not in

7) Which operators have got lowest priority ? ---> Logical operators i.e. and , or , not

```
# ++ and -- operators demo program
a = 25
print(++a) # +(+a) = +a = 25
#print(a++) # (a+)+ = a+ = 25+ throws error
print(a++1) # a + (+1) = 25 + 1 = 26
print(--a) # -(-a) = +a = 25
#print(a--) # (a-)- = a+ = 25+ throws error
print(a--1) # a - (-1) = a + 1 = 26
print(-a) # -25
print(+-a) # +(-a) = -a = -25
print(-+a) # -(+a) = -a = -25
'''
1) How to increment an object ? ---> obj += 1 (or) obj = obj + 1
2) How to decrement an object ? ---> obj -= 1 (or) obj = obj - 1
3) There are no ++ and -- operators in python
'''
```

```
# Semicolon demo program
print('One'); # ; is optional
print('Two'); # ; is optional
print('Three'); # ; is optional
print('Hyd') ; print('Sec') ; print('Cyb') # ; is mandator becoz multiple statements are in same line
'''
1) When is ; mandatory ? ---> When multiple statements are in the same line
   When is ; optional ? ---> When each statement is in a different line
2) print('Hyd') ; print('Sec') ; print('Cyb')
    Can comma be used instead of semicolon ? ---> Yes
                                                    i.e. print('Hyd') , print('Sec') , print('Cyb')
3) How many lines of output is generated by the above program ? ---> 6
4) What does print('Hyd') do ? ---> Prints Hyd and automatically moves to next line
'''
```