operators

arithemetic operators

```
In [3]: 28+45
```

```
Out[3]: 73
```

```
In [4]: 28-13
```

```
Out[4]: 15
```

```
In [5]: 27*5
```

```
Out[5]: 135
```

```
In [6]: 25//5
```

```
Out[6]: 5
```

assignment operators

```
In [8]: x=5
        x
```

```
Out[8]: 5
```

```
In [9]: x+=2
        x
```

```
Out[9]: 7
```

```
In [10]: x*=2
```

```
In [11]: x
```

```
Out[11]: 14
```

```
In [22]: x-=2
         x
```

```
Out[22]: 12
```

```
In [24]: x//=2
         x
```

```
Out[24]: 6
```

relation operators

```
In [27]: a=3
         b=5
```

```
In [29]: a<b
```

```
Out[29]: True
```

```
In [31]: b>a
```

```
Out[31]: True
```

```
In [33]: a==b
```

```
Out[33]: False
```

```
In [35]: a=5
```

```
In [37]: a==b
```

```
Out[37]: True
```

logical operator

```
In [40]: x=8
         y=4
```

```
In [42]: x>y and x>y # and table 1+1=1,0+0=0,1+0=0,0+1=0
```

```
Out[42]: True
```

```
In [46]: x==y and y==x
```

```
Out[46]: False
```

```
In [52]: y<x and y>x
```

```
Out[52]: False
```

```
In [54]: y>x and x>y
```

```
Out[54]: False
```

```
In [56]: x>y or x>y # or table 1+1=1,0+0=0,0+1=1,1+0=1
```

```
Out[56]: True
```

```
In [58]: x==y or y==x
```

```
Out[58]: False
```

In [60]:
```
y>x or y<x
```
Out[60]:
```
True
```

In [62]:
```
y<x or y>x
```
Out[62]:
```
True
```

In [66]:
```
not x>y # not will give the inverted answer
```
Out[66]:
```
False
```

unary operator

In [124…
```
n=5 #unary is negation of a number
```

In [126…
```
n
```
Out[126…
```
5
```

In [128…
```
n=-n
n
```
Out[128…
```
-5
```

In [132…
```
n1=-54
n1
```
Out[132…
```
-54
```

In [134…
```
n1=-n1
n1
```
Out[134…
```
54
```

In [ ]:
```

```

swap

In [69]:
```
a=5
b=10
```

In [71]:
```
w=a # swapping using three variable
a=b
b=w
```

In [73]:
```
print (a)
print(b)
```
```
10
5
```

In [75]:
```
a1=6
b1=9
```

In [77]:
```
a1,b1=b1,a1 #rot swapping
print(a1)
print(b1)
```

```
9
6
```

numbers system

In [80]:
```
bin(10) # in binary numbers system number lies between 0 and 1
```

Out[80]:
```
'0b1010'
```

In [82]:
```
0b010110
```

Out[82]:
```
22
```

In [84]:
```
oct(20) #in octal number system numbers lies between 0 to 7
```

Out[84]:
```
'0o24'
```

In [86]:
```
0o31
```

Out[86]:
```
25
```

In [88]:
```
hex(15) #in hexdecimal number system number starts with 0 and goes till 9 and af
```

Out[88]:
```
'0xf'
```

In [90]:
```
0x1a
```

Out[90]:
```
26
```

bitwise operator

compliment

In [95]:
```
~35 # compliment converts the 0 s to 1 s and 1 s to 0 s of a binary form of a nu
```

Out[95]:
```
-36
```

In [97]:
```
~82
```

Out[97]:
```
-83
```

and

In [100…
```
25&33 #and-will compare binary form of both numbers and give new binary form of
```
Out[100…
```
1
```
In [102…
```
34&12
```
Out[102…
```
0
```
In [104…
```
2&3
```
Out[104…
```
2
```
In [106…
```
35&40
```
Out[106…
```
32
```

or

In [109…
```
32|33 #or-will compare binary form of both numbers and give new binary form of t
```
Out[109…
```
33
```
In [111…
```
2|4
```
Out[111…
```
6
```
In [113…
```
43|21
```
Out[113…
```
63
```

xor

In [116…
```
20^24 #xor-will compare binary form of both numbers and give new binary form of
```
Out[116…
```
12
```
In [118…
```
16^55
```
Out[118…
```
39
```
In [120…
```
2^5
```
Out[120…
```
7
```

left shift

In [155…
```
0b10100
```
Out[155…
```
20
```

In [157…
```
0b1010000
```
Out[157…
```
80
```
In [138…
```
20<<2 #left shift will add zeros at the end of the binary form of number and giv
      #binary form of 20 is 10100 and added two zeros at the end it gives 101000
```
Out[138…
```
80
```
In [140…
```
2<<1 # 2 in binary is 10 after adding 00 i.e 100 is 4
```
Out[140…
```
4
```
In [142…
```
43<<3
```
Out[142…
```
344
```

right shift

In [145…
```
0b1010
```
Out[145…
```
10
```
In [147…
```
0b10
```
Out[147…
```
2
```
In [149…
```
10>>2 #left shift will remove the 1s and 0s at the end of the binary form of num
      #binary form of 10 is 1010 and removed 1,0 at the end it gives 10 i.e 2
```
Out[149…
```
2
```
In [151…
```
23>>4 #23  in binary is 10111 after removing 0111 it gives 10 i.e 2
```
Out[151…
```
1
```
In [153…
```
38>>1
```
Out[153…
```
19
```

input function

In [160…
```
a=input('enter first number') #input always prints results in string data type
b=input('enter second number')
c=a+b
print(c)
```
```
35
```
In [162…
```
a=int(input('enter first number')) # we can convert the string into integer befo
b=int(input('enter second number'))
c=a+b
print(c)
```

```
86
```

In [168…

```
result=input('enter exp') #the expression will be printed in the string data typ
result
```

Out[168…

```
'10+5-4*2'
```

In [170…

```
result=eval(input('enter exp')) #by using eval function we can solve the express
result
```

Out[170…

```
7
```

In [172…

```
char=input('enter the char')
char
```

Out[172…

```
'hello python'
```

In [174…

```
char=input('enter the char')[2] # this will index the string which will print
char
```

Out[174…

```
'l'
```

In [176…

```
char=input('enter the char')[0:5] # this will slice the string which will print
char
```

Out[176…

```
'hello'
```

In [ ]:

In [ ]:

In [ ]: