# operators

#### 1) Arithemetic operators(+,-,,/,%,%%,*,^)

#### 2) Assignment operators ¶

#### 3) Relational operators

#### 4) logical operators

#### 5) Unary operator

## 1) Arithemetic operator

```
In [1]: x1,y1=10,5
```

```
In [2]: x1+y1
Out[2]: 15
```

```
In [3]: x1*y1
Out[3]: 50
```

```
In [4]: x1/y1
Out[4]: 2.0
```

```
In [5]: x1//y1
Out[5]: 2
```

```
In [6]: x1%y1
Out[6]: 0
```

```
In [7]: x1**y1
Out[7]: 100000
```

```
In [8]: x2=3
        y2=3
```

```
In [9]: x2**y2
Out[9]: 27
```

## Assignment operator

```
In [10]: x=2

In [11]: x=x+2

In [12]: x
Out[12]: 4

In [13]: x+=2
         x
Out[13]: 6

In [14]: x=+2

In [15]: x
Out[15]: 2

In [16]: x=+3

In [17]: x
Out[17]: 3

In [18]: x+=3

In [19]: x
Out[19]: 6

In [20]: x*=2

In [21]: x
Out[21]: 12
```

In [22]:
```
x-=3
x
```
Out[22]:
```
9
```

In [23]:
```
x/=3
```

In [24]:
```
x
```
Out[24]:
```
3.0
```

In [25]:
```
x//=1
x
```
Out[25]:
```
3.0
```

In [26]:
```
x//=3
```

In [27]:
```
x
```
Out[27]:
```
1.0
```

In [28]:
```
a,b=5,6
print(a)
print(b)
```
```
5
6
```

In [29]:
```
a=5
b=6
print(a)
print(b)
```
```
5
6
```

In [30]:
```
a
```
Out[30]:
```
5
```

In [31]:
```
b
```
Out[31]:
```
6
```

In [32]:
```
a,b
```
Out[32]:
```
(5, 6)
```

## unary operator

**unary means 1|| binary means 2**

**here we are applying unary minus operator(-) on the operand n; the value of m becomes -7, which indicates it as a negative value**

```
In [33]: n=7
         n

Out[33]: 7

In [34]: m=-(n)
         m

Out[34]: -7

In [35]: n

Out[35]: 7

In [36]: -n

Out[36]: -7
```

## Relational operator

```
In [37]: ## we are using this operator for comparing

In [38]: a=5
         b=6

In [39]: a<b

Out[39]: True

In [40]: a>b

Out[40]: False

In [41]: a=b # we cannot use = operator that means it is assigning

In [42]: a

Out[42]: 6
```

```
In [43]: a==b
Out[43]: True
```

```
In [44]: a!=b
Out[44]: False
```

```
In [45]: a=7
```

```
In [46]: a
Out[46]: 7
```

```
In [47]: b
Out[47]: 6
```

```
In [48]: a>b
Out[48]: True
```

```
In [49]: a<b
Out[49]: False
```

```
In [50]: a<=b
Out[50]: False
```

```
In [51]: a!=b
Out[51]: True
```

## LOGICAL OPERATOR

```
In [52]: ## AND,OR,NOT
```

```
In [53]: a=5
         b=4
```

```
In [54]: a<8 and b<2
Out[54]: False
```

```
In [55]:  a<8 or b<2

Out[55]:  True

In [56]:  a>8 or b<2

Out[56]:  False

In [57]:  x=False
          x

Out[57]:  False

In [58]:  not x      # you can reverse the operation

Out[58]:  True

In [59]:  x=not x
          x

Out[59]:  True
```

## Number system coversation(bit-binary digit)

```
In [60]:  25

Out[60]:  25

In [61]:  bin(25)

Out[61]:  '0b11001'

In [62]:  0b11001

Out[62]:  25

In [63]:  int(0b11001)

Out[63]:  25

In [64]:  bin(35)

Out[64]:  '0b100011'

In [65]:  int(0b100011)

Out[65]:  35
```

```
In [66]: bin(20)
Out[66]: '0b10100'

In [67]: int(0b10100)
Out[67]: 20

In [68]: 0b1111
Out[68]: 15

In [69]: oct(15)
Out[69]: '0o17'

In [70]: 0o17
Out[70]: 15

In [71]: hex(9)
Out[71]: '0x9'

In [72]: 0xf
Out[72]: 15

In [73]: hex(10)
Out[73]: '0xa'
```

## number system prefix in python

#### binary---->0b---base 2

#### octal----->0o----base 8

#### hexadecimal-->0x----base 16

```
In [74]: hex(25)
Out[74]: '0x19'

In [75]: hex(27)
Out[75]: '0x1b'
```

## swap variable in python

#### (a,b=5,6) After swap we should get==>(a,b=6,5)

In [76]:
```
a=5
b=6
```

In [77]:
```
a=b
b=a
```

In [78]:
```
a,b
```

Out[78]:
```
(6, 6)
```

In [79]:
```
a,b=b,a
```

In [80]:
```
print(a)
```

```
6
```

In [81]:
```
print(b)
```

```
6
```

In [82]:
```
a1=7
b1=8
```

In [83]:
```
temp=a1
a1=b1
b1=temp
```

In [84]:
```
print(a1)
print(b1)
```

```
8
7
```

In [85]:
```
a2=5
b2=6
```

In [86]:
```
# swap variable formulas
a2=a2+b2
b2=a2-b2
a2=a2-b2
```

In [87]:
```
print(a2)
print(b2)
```
```
6
5
```

In [88]:
```
print(0b101)
print(0b110)
```
```
5
6
```

In [89]:
```
print(0b1011)
```
```
11
```

In [90]:
```
a2=a2^b2
b2=a2^b2
a2=a2^b2
```

In [91]:
```
print(a2)
print(b2)
```
```
5
6
```

In [92]:
```
a2,b2=b2,a2
```

In [93]:
```
print(a2)
print(b2)
```
```
6
5
```

# BITWISE OPERATOR

**we have 6 opertors**

***COMPLIMENT(~)***

***AND(&)***

***OR(|)***

***XOR(^)***

***LIFT SHIFT(<<)***

***RIGHT SHIFT(>>)***

```
In [94]: ~2
Out[94]: -3

In [95]: ~12
Out[95]: -13

In [96]: ~45
Out[96]: -46

In [97]: ~6
Out[97]: -7

In [98]: ~1
Out[98]: -2
```

## Bitwise AND operator

```
In [99]: 12&13
Out[99]: 12

In [100]: 1&1
Out[100]: 1

In [101]: 1|0
Out[101]: 1

In [102]: 1&0
Out[102]: 0

In [103]: 12|13
Out[103]: 13

In [104]: bin(35)
Out[104]: '0b100011'
```

```
In [105]: 12^13
Out[105]: 1

In [106]: 25^30
Out[106]: 7

In [107]: bin(25)
Out[107]: '0b11001'

In [108]: bin(30)
Out[108]: '0b11110'

In [109]: int(100011)
Out[109]: 100011

In [110]: bin(100011)
Out[110]: '0b11000011010101011'

In [111]: int(0b000111)
Out[111]: 7

In [112]: bin(5)
Out[112]: '0b101'

In [113]: 5<<2
Out[113]: 20

In [114]: 5<<3
Out[114]: 40

In [115]: (101000)
Out[115]: 101000

In [116]: 5>>2
Out[116]: 1
```

```
In [117]:  9>>2

Out[117]: 2

In [118]:  9<<2

Out[118]: 36
```

## import math module

```
In [119]:  import math as m

In [120]:  x=sqrt(25)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-120-2d0b08ab89cc> in <module>
----> 1 x=sqrt(25)

NameError: name 'sqrt' is not defined


In [121]:  x=math.sqrt(5)
           x

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-121-df932bf73265> in <module>
----> 1 x=math.sqrt(5)
      2 x

NameError: name 'math' is not defined


In [122]:  x=m.sqrt(5)
           x

Out[122]: 2.23606797749979

In [123]:  x=m.sqrt(25)
           x

Out[123]: 5.0

In [124]:  print(m.floor(2.9))  #print the minimum or least value

2
```

In [125]:
```
print(m.ceil(2.9))
```
```
3
```

In [126]:
```
print(m.pow(3,2))
```
```
9.0
```

In [127]:
```
print(m.pi)
```
```
3.141592653589793
```

## user input function in python || command line input

In [128]:
```
x1=input('Enter the 1 st number')
y1=input('Enter the second number')
z1=x1+y1
print(z1)
```
```
Enter the 1 st number5
Enter the second number6
56
```

In [129]:
```
x=input()
y=input()
z=x+y
print(z)
```
```
'akshitha'
'perumandla'
'akshitha''perumandla'
```

In [130]:
```
x=input()
y=input()
z=x+y
print(z)
```
```
1
2
12
```

#### when you works on input function it always gives you a string

In [132]:
```
type(x)
type(y)
```
Out[132]:
```
str
```

In [133]:
```
x1=input('Enter the 1st number')
a1=int(x1)
y1=input('Enter the 2nd number')
b1=int(y1)
z1=a1+b1
print(z1)
```
```
Enter the 1st number1
Enter the 2nd number2
3
```

In [134]:
```
x2=int(input('Enter the 1st number'))
y2=int(input('Enter the 2nd number'))
z2=x2+y2
print(z2)
```
```
Enter the 1st number12
Enter the 2nd number5
17
```

In [135]:
```
ch=input('enter a char')
print(ch)
```
```
enter a char1
1
```

In [136]:
```
ch=input('enter a char')
print(ch)
```
```
enter a charakshitha
akshitha
```

In [137]:
```
print(ch[0])
```
```
a
```

In [138]:
```
print(ch[-1])
```
```
a
```

In [139]:
```
ch=input('enter a char')[0]
print(ch)
```
```
enter a charakshitha
a
```

In [140]:
```
ch=input('enter a char')[1:3]
print(ch)
```
```
enter a charakshitha
ks
```

## EVAL function using input

In [143]:
```
result=eval(input('enter an expr'))
print(result)
```

```
enter an expr12
12
```

In [ ]: