```
In [77]: 1+1 # ADDITION

Out[77]: 2

In [78]: 2-1

Out[78]: 1

In [79]: 3*4

Out[79]: 12

In [80]: 8/4 # Division

Out[80]: 2.0

In [81]: 8/5 # Float division

Out[81]: 1.6

In [82]: 8/4

Out[82]: 2.0

In [83]: 8 // 4 # integer division

Out[83]: 2

In [84]: 8 + 9 -7

Out[84]: 10

In [85]: 8 + 8 -

  Cell In[85], line 1
    8 + 8 -
           ^
SyntaxError: invalid syntax

In [86]: 5 + 5 * 5 # BODMAS (Bracket || Oders || Divide || Multiply || Add || Substact)

Out[86]: 30

In [87]: 2 * 2 * 2 * 2 * 2 * 2 # exponentation

Out[87]: 64

In [88]: 2 * 5

Out[88]: 10

In [89]: 2 ** 5

Out[89]: 32
```

```
In [90]: 15 / 3

Out[90]: 5.0

In [91]: 10 // 3

Out[91]: 3

In [92]: 15 % 2 # Modulus

Out[92]: 1

In [93]:  10 % 2

Out[93]: 0

In [94]: 15 %% 2

  Cell In[94], line 1
    15 %% 2
        ^
SyntaxError: invalid syntax

In [95]:  3 + 'nit'

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[95], line 1
----> 1 3 + 'nit'

TypeError: unsupported operand type(s) for +: 'int' and 'str'

In [96]: a,b,c,d,e = 15,7.8,'nit',8+9j,True
         print(a)
         print(b)
         print(c)
         print(d)
         print(e)

15
7.8
nit
(8+9j)
True

In [97]: print(type(a))
         print(type(b))
         print(type(c))
         print(type(d))
         print(type(e))

<class 'int'>
<class 'float'>
<class 'str'>
<class 'complex'>
<class 'bool'>

In [98]:  type(c)
```

Out[98]:
```
str
```

string

In [99]:
```
'Naresh IT'
```

Out[99]:
```
'Naresh IT'
```

In [100…
```
print('Max it')
```

```
Max it
```

In [101…
```
"max it technology"
```

Out[101…
```
'max it technology'
```

In [102…
```
s1 = 'max it technology'
s1
```

Out[102…
```
'max it technology'
```

In [103…
```
a = 2
b = 3
a + b
```

Out[103…
```
5
```

In [104…
```
c = a + b
c
```

Out[104…
```
5
```

In [105…
```
a = 3
b = 'hi'
type(b)
```

Out[105…
```
str
```

In [106…
```
print('max it's"Technology"')  # \ has some special meaning to ignore the error
```

```
  Cell In[106], line 1
    print('max it's"Technology"')  # \ has some special meaning to ignore the err
or
                                ^
SyntaxError: unterminated string literal (detected at line 1)
```

In [107…
```
print('max it\'s"Technology"') #\ has some special meaning to ignore the error
```

```
max it's"Technology"
```

In [108…
```
print('max it',  'Technology')
```

```
max it Technology
```

In [109…
```
'nit' + ' nit'
```

```
Out[109… 'nit nit'

In [110… 'nit'  ' nit'

Out[110… 'nit nit'

In [111… 5 * 'nit'

Out[111… 'nitnitnitnitnit'

In [112… 5*' nit' # space between words

Out[112… ' nit nit nit nit nit'

In [113… print('c:\nit') #\n -- new line

c:
it

In [114… print(r'c:\nit') #raw string

c:\nit
```

variable

```
In [115…  2

Out[115… 2

In [116… x = 2
         x

Out[116… 2

In [117… x + 3

Out[117… 5

In [118… y = 3
         y

Out[118… 3

In [119…  x + y

Out[119… 5

In [120… x = 9
         x

Out[120… 9

In [121… x + y

Out[121… 12
```

```
In [122…  x + 10

Out[122…  19

In [123…  _ + y

Out[123…  22

In [124…  _ + y

Out[124…  25

In [125…  _ + y

Out[125…  28

In [126…  y

Out[126…  3

In [127…  # string variable
          name = 'mit'
          name

Out[127…  'mit'

In [128…  name + 'technology'

Out[128…  'mittechnology'

In [129…  name 'technology'

            Cell In[129], line 1
              name 'technology'
                   ^
          SyntaxError: invalid syntax

In [130…  name

Out[130…  'mit'

In [131…  len(name)

Out[131…  3

In [132…  name[0]

Out[132…  'm'

In [133…  name[5]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[133], line 1
----> 1 name[5]

IndexError: string index out of range
```

In [134…
```
name[7]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[134], line 1
----> 1 name[7]

IndexError: string index out of range
```

In [135…
```
name[-1]
```

Out[135…
```
't'
```

In [136…
```
name[-2]
```

Out[136…
```
'i'
```

In [137…
```
name[-6]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[137], line 1
----> 1 name[-6]

IndexError: string index out of range
```

slicing

In [138…
```
name
```

Out[138…
```
'mit'
```

In [139…
```
name[0:1]
```

Out[139…
```
'm'
```

In [140…
```
name[1:4]
```

Out[140…
```
'it'
```

In [141…
```
name[1:]
```

Out[141…
```
'it'
```

In [142…
```
name[:4]
```

Out[142…
```
'mit'
```

```
In [143… name[3:9]

Out[143… ''

In [144… name

Out[144… 'mit'

In [145… name1 = 'fine'
         name1

Out[145… 'fine'

In [146… name1[0:1]

Out[146… 'f'

In [147… name1[0:1] = 'd'

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[147], line 1
----> 1 name1[0:1] = 'd'

TypeError: 'str' object does not support item assignment

In [148… name1[0] = 'd'

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[148], line 1
----> 1 name1[0] = 'd'

TypeError: 'str' object does not support item assignment

In [149… name1

Out[149… 'fine'

In [150… name1[1:]

Out[150… 'ine'

In [151… 'd' + name1[1:] #i want to change fine to dine

Out[151… 'dine'

In [152… num1.insert(2,''nit')

  Cell In[152], line 1
    num1.insert(2,''nit')
                       ^
SyntaxError: unterminated string literal (detected at line 1)
```

Introduce to ID()

```
In [154…  num = 5 # variable address
          id(num)

Out[154…  140729894906424

In [155…  name = 'nit'
          id(name) # Address will be different for both

Out[155…  1841196533344

In [156…  a = 10
          id(a)

Out[156…  140729894906584

In [157…  b = a # thats why python is more efficient

In [158…  id(b)

Out[158…  140729894906584

In [159…  id(10)

Out[159…  140729894906584

In [160…  a = 20
          id(a)# as we change the value of then address will also change

Out[160…  140729894906904

In [161…  id(b)

Out[161…  140729894906584

In [162…  PI = 3.14 # In maths pi value is constant but in python we change pi value
          PI

Out[162…  3.14

In [163…  PI = 3.15
          PI

Out[163…  3.15

In [164…  type(PI)

Out[164…  float
```

Arithmetic Operator

```
In [165…  x1, y1 = 10,5

In [167…   x1 ^ y1
```

```
Out[167… 15

In [168… x1 + y1

Out[168… 15

In [169… x1 - y1

Out[169… 5

In [170… x1 * y1

Out[170… 50

In [171… x1 / y1

Out[171… 2.0

In [172… x1 // y1

Out[172… 2

In [173… x1 % y1

Out[173… 0

In [174… x1 ** y1

Out[174… 100000
```

Assignment operator

```
In [175… x = 2

In [176… x = x+2

In [177… x

Out[177… 4

In [179… x + = 2
         x
```

```
  Cell In[179], line 1
    x + = 2
        ^
SyntaxError: invalid syntax
```

```
In [180… x * = 2
         x
```

```
  Cell In[180], line 1
    x * = 2
        ^
SyntaxError: invalid syntax
```

In [181…
```
x - = 2
x
```

```
  Cell In[181], line 1
    x - = 2
        ^
SyntaxError: invalid syntax
```

In [182…
```
x / = 2
x
```

```
  Cell In[182], line 1
    x / = 2
        ^
SyntaxError: invalid syntax
```

In [183…
```
a,b = 5,6
```

In [184…
```
a
```

Out[184…
```
5
```

In [185…
```
b
```

Out[185…
```
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 [186…
```
n = 7
n
```

Out[186…
```
7
```

In [187…
```
m = -(n)
m
```

Out[187…
```
-7
```

In [188…
```
n
```

Out[188…
```
7
```

In [189…
```
-n
```

Out[189…
```
-7
```

Relational operator

we are using this operator for comparing

```
In [190…  a = 5
          b = 6

In [191…  a < b

Out[191…  True

In [192…  a > b

Out[192…  False

In [193…  # a = b # we cannot use = operatro that means it is assigning

In [194…  a == b

Out[194…  False

In [196…  a != b

Out[196…  True

In [197…  b = 5

In [198…  b

Out[198…  5

In [199…  a == b

Out[199…  True

In [200…  a >= b

Out[200…  True

In [201…  a <=b

Out[201…  True

In [202…  a < b

Out[202…  False

In [203…  a > b

Out[203…  False

In [204…  b = 7
In [205…  a != b

Out[205…  True
```

Logical operator

- AND,OR,NOT

```
In [206…  a = 5
          b = 4

In [207…  a < 8 and b < 5

Out[207…  True

In [208…  a < 8 and b < 2

Out[208…  False

In [209…  a > 8 or b < 2

Out[209…  False

In [210…  x = False
          x

Out[210…  False

In [211…  not x

Out[211…  True
```

Number system converstion (bit-binary digit)

```
In [1]:  25

Out[1]:  25

In [2]:  bin(25)

Out[2]:  '0b11001'

In [3]:  0b11001

Out[3]:  25

In [4]:  int(0b11001)

Out[4]:  25

In [5]:  bin(20)
```

```
Out[5]:  '0b10100'

In [6]:  int(0b10100)

Out[6]:  20

In [7]:  oct(15)

Out[7]:  '0o17'

In [8]:  0o17

Out[8]:  15

In [9]:  hex(9)

Out[9]:  '0x9'

In [10]: 0xf

Out[10]: 15

In [11]: hex(10)

Out[11]: '0xa'

In [12]: 0xa

Out[12]: 10

In [13]: hex(25)

Out[13]: '0x19'

In [14]: 0x19

Out[14]: 25

In [15]: 0x15

Out[15]: 21
```

swap variable in python

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

```
In [19]: a = 5
         b = 6

In [20]: a = b
         b = a

In [21]: a,b = b,a
In [22]: print(a)
         print(b)
```

```
6
6
```

```
In [23]: a1 = 7
         b1 = 8
```

```
In [24]: temp = a1
         a1 = b1
         b1 = temp
```

```
In [25]: print(a1)
         print(b1)
```

```
8
7
```

```
In [26]: a2 = 5
         b2 = 6
```

```
In [28]: a2 = a2 + b2 # swap variable formulas
         b2 = a2 - b2
         a2 = b2 - b2
```

```
In [29]: print(a2)
         print(b2)
```

```
0
0
```

```
In [30]: print(0b101)  # 101 is 3 bit
         print(0b110)  # 110 is 3 bit
```

```
5
6
```

```
In [42]: print(bin(11))
         print(0b1011) # Here it will get 4 bit
```

```
0b1011
11
```

```
In [41]: a2 = a2 ^ b2
         b2 = a2 ^ b2 # using XOR to swap variable because it will not waste extra bit
         a2 = a2 ^ b2
```

```
In [33]: print(a2)
         print(b2)
```

```
0
0
```

```
In [43]: print(a2)
         print(b2)
```

```
0
0
```

In [44]:
```
a2 , b2 = b2 , a2
```

In [45]:
```
print(a2)
print(b2)
```

```
0
0
```

BITWISE OPERATOR

- WE HAVE 6 OPERATORS

COMPLEMENT (~) || AND (&) || OR (|) || XOR (^) || LEFT SHIFT (<<) || RIGHT SHIFT (>>)

Complement (~)

In [47]:
```
~12
```

Out[47]:
```
-13
```

In [48]:
```
~45
```

Out[48]:
```
-46
```

In [49]:
```
~6
```

Out[49]:
```
-7
```

In [50]:
```
~-6
```

Out[50]:
```
5
```

In [51]:
```
~-1
```

Out[51]:
```
0
```

And (&)

In [53]:
```
12 & 13
```

Out[53]:
```
12
```

In [54]:
```
1 & 1
```

Out[54]:
```
1
```

In [55]:
```
1 | 0
```

Out[55]:
```
1
```

In [56]:
```
1 & 0
```

```
Out[56]: 0
```

```
In [57]: 12 | 13
```

```
Out[57]: 13
```

```
In [58]: 35 & 40
```

```
Out[58]: 32
```

```
In [59]: 35 | 40
```

```
Out[59]: 43
```

XOR (^)

```
In [61]: 12 ^ 13
```

```
Out[61]: 1
```

```
In [62]: 25 ^ 30
```

```
Out[62]: 7
```

```
In [63]: bin(25)
```

```
Out[63]: '0b11001'
```

```
In [64]: bin(30)
```

```
Out[64]: '0b11110'
```

```
In [65]: int(0b000111)
```

```
Out[65]: 7
```

Left Operator (<<)

```
In [66]: 10 << 2
```

```
Out[66]: 40
```

```
In [68]: 20<<4
```

```
Out[68]: 320
```

```
In [69]: bin(20)
```

```
Out[69]: '0b10100'
```

Right Shift (>>)

```
In [70]: 10>>2
```

```
Out[70]:  2

In [71]:  bin(20)

Out[71]:  '0b10100'

In [72]:  20>>4

Out[72]:  1
```

import math module

```
In [73]:  x = sqrt(25)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[73], line 1
----> 1 x = sqrt(25)

NameError: name 'sqrt' is not defined

In [74]:  import math

In [76]:  x = math.sqrt(25)
          x

Out[76]:  5.0

In [77]:  x1 = math.sqrt(15)
          x1

Out[77]:  3.872983346207417

In [78]:  print(math.floor(2.9)) # floor = minimum or least value

         2

In [79]:  print(math.ceil(2.9)) # ceil - maximum or heighest value

         3

In [80]:  print(math.pow(3,2))

         9.0

In [81]:  print(math.pi)

         3.141592653589793

In [82]:  print(math.e)

         2.718281828459045

In [83]:  import math as m
          m.sqrt(10)

Out[83]:  3.1622776601683795
```

```
In [84]: from math import sqrt,pow
         pow(2,3)

Out[84]: 8.0

In [85]: round(pow(2,3))

Out[85]: 8

In [1]: x = input()
        y = input()
        z = x + y
        print(z)

        98

In [2]: x1 = input('Enter the 1st number')
        y1 = input('Enter the 2nd number')
        z1 = x1 + y1
        print(z1)

        78

In [3]: type(x1)
        type(y1)

Out[3]: str

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

        16

In [1]: x2 = int(input('Enter the 1st number'))
        y2 = int(input('Enter the 2nd number'))
        z2 = x2 + y2
        z2

Out[1]: 17

In [2]: ch = input('enter a char')
        print(ch)

        NIT

In [3]: print(ch[0])

        N

In [4]: print(ch[1])

        I

In [5]: print(ch[-1])

        T
```

```
In [1]: ch = input('enter a char')[0]
        print(ch)
```

```
g
```

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

```
op
```

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

```
In [ ]:
```