```
x= 1 #store value => integer value /datatype /datastructure
y=2.5 #float
sum=x+y
print(sum)
```

```
3
```

```
x= 1 #store value => integer value /datatype /datastructure
y=2.5 #float
x='ram' #string=> single/double quotation
x=5j #complex
sum=x+y
print(sum)
```

```
(2.5+5j)
```

```
a =5 #integer
print(type(a))
```

```
<class 'int'>
```

```
print (id(a))
```

```
140709148375608
```

```
name = "my name is garima. I am from kathmandu " # my name is garima ra I am from kathmandu lai double line ma lekhna meldeana yo single ra d
print(name)
```

```
my name is garima. I am from kathmandu
```

```
name ="""my name is Garima rokaha.
I am from kathmandu"""
print (name)
```

```
my name is Garima rokaha.
I am from kathmandu
```

```
#boolean
x = True
print(x,type(x))
```

```
True <class 'bool'>
```

```
#None datatype
x= None
print(x, type(x))
```

```
None <class 'NoneType'>
```

```
#list datatype
x=[1,2,3,4]
print(x, type(x))
```

```
[1, 2, 3, 4] <class 'list'>
```

```
#tuple datatype
x =(1,2,3,4)
print(x,type(x))
```

```
(1, 2, 3, 4) <class 'tuple'>
```

```
#set datatype
x={1,2,3,4}
print(x,type(x))
```

```
{1, 2, 3, 4} <class 'set'>
```

```
# dictionary
x= {
    'name': 'ram', #key:value pair with coma seperated
    "age" : 22,
    "address" : 'ktm', # last coma is optonal
}
print(x , type(x))
```

```
{'name': 'ram', 'age': 22, 'address': 'ktm'} <class 'dict'>
```

```
x = input("Enter the value of x:")
y = input("Enter the value of y:")
sum= x+y
print(sum)
```

```
Enter the value of x: 23
Enter the value of y: 45
2345
```

```
x =  int (input("Enter the value of x:"))
y =  int (input("Enter the value of y:"))
sum= x+y
print( f"sum of {x} and {y} is {sum}")
```

```
Enter the value of x: 2
Enter the value of y: 3
sum of 2 and 3 is 5
```

```
#operator
2+3 #addition
5-6 #substraction
5*4 #multiplication
2**5 #for power (square)
2**5 #for cube
5/2.5 #division
5//2 # division opearator in tbe o/p
5%2 #modulus
```

```
1
```

```
name ='ramesh'
len(name)
```

```
6
```

```
#precedence
#small bracket
#and
#or
```

```
len('apple')==len('banana')
```

```
False
```

```
False or True and False
#false or false
#false
```

```
False
```

```
#assignment operator
x=5
```

```
x=2
y=2
print (id(x))
print(id(y))
x is y # memory address>= identity operator (same memory address am cha ki nai check garna is use hunchha)
```

```
140709148375512
140709148375512
```

```
True
```

```
#membership operator
1 in [1,2,3,4,5,6]
```

```
True
```

```
'a' in ['A','b',2,3]
```

```
False
```

```
'r' in 'ramesh'
```

```
True
```

```
're ' in 'ramesh'
```

```
False
```

```
'a' in ('A','b',2,3)
```

```
False
```

```
#17 number after decimal point
b=0.1234567890123456789
print(b)
```

```
0.12345678901234568
```

```
(1.1+2.2)==3.3
```

```
False
```

```
1.1+2.2
```

```
3.3000000000000003
```

```
from decimal import Decimal as D #alias
print (D('1.1')+D('2.2'))
```

```
3.3
```

```
from math import pi
print (pi)
```

```
3.141592653589793
```

```
# math , decimal are modules in python
```

```
import math
print(math.pi)
print(math.cos(math.pi)) #cos(pi)= -1
print(math.exp(10))
print(math.log10(1000)) #log10(1000)=3
```

```
3.141592653589793
-1.0
22026.465794806718
3.0
```

```
help (math)
```

```
Help on built-in module math:

NAME
    math
    DESCRIPTION
        This module provides access to the mathematical functions
        defined by the C standard.

    FUNCTIONS
        acos(x, /)
            Return the arc cosine (measured in radians) of x.

            The result is between 0 and pi.

        acosh(x, /)
            Return the inverse hyperbolic cosine of x.

        asin(x, /)
            Return the arc sine (measured in radians) of x.

            The result is between -pi/2 and pi/2.

        asinh(x, /)
            Return the inverse hyperbolic sine of x.

        atan(x, /)
            Return the arc tangent (measured in radians) of x.

            The result is between -pi/2 and pi/2.

        atan2(y, x, /)
            Return the arc tangent (measured in radians) of y/x.

            Unlike atan(y/x), the signs of both x and y are considered.

        atanh(x, /)
            Return the inverse hyperbolic tangent of x.

        cbrt(x, /)
            Return the cube root of x.

        ceil(x, /)
            Return the ceiling of x as an Integral.

            This is the smallest integer >= x.

        comb(n, k, /)
            Number of ways to choose k items from n items without repetition and without order.

            Evaluates to n! / (k! * (n - k)!) when k <= n and evaluates
            to zero when k > n.

            Also called the binomial coefficient because it is equivalent
            to the coefficient of k-th term in polynomial expansion of the
            expression (1 + x)**n.
```

```
s="This is string."
print(s,type(s))
```

```
This is string. <class 'str'>
```

```
s="This is string."
print(s[5])
```

```
i
```

```
s="This is string."
s[0:4]#slicing
```

```
'This'
```

```
s="This is string."
print(s[0],s[1],s[2],s[3])
```

```
T h i s
```

```
s="This is string. strings in python are immutable." #e 46 ma raichhaa so s[46] lekhelo e display garauna ko lagi
s[46]
```

```
'e'
```

```
s="This is string. strings in python are immutable."
s[-2] #negative indexing
```

```
'e'
```

Start coding or generate with AI.