In [ ]:
```
let assume that  our friend coder is there

he created  an addition program

he feels that he only created an addition program

he wants to give this program to every one ===== Package

he wents to anaconda organization

he makes a deal

who ever installed anaconda my package also automatically download

thats why when you are downlaoding anaconda a green color pop is comes
means so many packages are tie with anaconda
downloading in your local laptop
```

In [ ]:
```
package is there in your laptop
you want to use for the coding

the package name: addition
```

import

In [ ]:
```
# syntax
#import <package_name>
```

In [9]:
```
#package name: random
import random
```

In [10]:
```
# package name: time
import time
```

In [11]:
```
# package name: math
import math
```

In [12]:
```
# package name: streamlit
import streamlit
```

In [ ]:
```
# Module not found
# package or module both are same
```

In [13]:
```
import cv2
```

In [ ]:
```
cv2 guy not tie with anaconda
no module name : cv2

streamlit guy not tie with anaconda
no module name: streamlit
```

In [ ]:
```
math operations
    addition
    subtraction
    multiplication
    division
```

In [14]:
```
import random
dir(random)
```

```
Out[14]: ['BPF',
          'LOG4',
          'NV_MAGICCONST',
          'RECIP_BPF',
          'Random',
          'SG_MAGICCONST',
          'SystemRandom',
          'TWOPI',
          '_ONE',
          '_Sequence',
          '_Set',
          '__all__',
          '__builtins__',
          '__cached__',
          '__doc__',
          '__file__',
          '__loader__',
          '__name__',
          '__package__',
          '__spec__',
          '_accumulate',
          '_acos',
          '_bisect',
          '_ceil',
          '_cos',
          '_e',
          '_exp',
          '_floor',
          '_index',
          '_inst',
          '_isfinite',
          '_log',
          '_os',
          '_pi',
          '_random',
          '_repeat',
          '_sha512',
          '_sin',
          '_sqrt',
          '_test',
          '_test_generator',
          '_urandom',
          '_warn',
          'betavariate',
          'choice',
          'choices',
          'expovariate',
          'gammavariate',
          'gauss',
          'getrandbits',
          'getstate',
          'lognormvariate',
          'normalvariate',
          'paretovariate',
          'randbytes',
          'randint',
          'random',
          'randrange',
          'sample',
          'seed',
 'setstate',
 'shuffle',
 'triangular',
 'uniform',
 'vonmisesvariate',
 'weibullvariate']
```

In [ ]:
```
# syntax
# <package_name>.<method_name>
# packagename: random
# methodname: randint
```

In [15]:
```
#help(<packagename.methodname>)
# help will give the understanding of
# how a method will wor

# here I want to know what randint will do ?

help(random.randint)
```

```
Help on method randint in module random:

randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.
```

In [19]:
```
random.randint(1,10)
```

Out[19]:
```
10
```

In [20]:
```
# step-1:  import <package_name>
# step-2:  dir(<pacakge_name>)
# step-3:  help(<pacakge_name>.<method_name>)
# package_name: random
import random
```

In [21]:
```
dir(random)
```

```
Out[21]:  ['BPF',
           'LOG4',
           'NV_MAGICCONST',
           'RECIP_BPF',
           'Random',
           'SG_MAGICCONST',
           'SystemRandom',
           'TWOPI',
           '_ONE',
           '_Sequence',
           '_Set',
           '__all__',
           '__builtins__',
           '__cached__',
           '__doc__',
           '__file__',
           '__loader__',
           '__name__',
           '__package__',
           '__spec__',
           '_accumulate',
           '_acos',
           '_bisect',
           '_ceil',
           '_cos',
           '_e',
           '_exp',
           '_floor',
           '_index',
           '_inst',
           '_isfinite',
           '_log',
           '_os',
           '_pi',
           '_random',
           '_repeat',
           '_sha512',
           '_sin',
           '_sqrt',
           '_test',
           '_test_generator',
           '_urandom',
           '_warn',
           'betavariate',
           'choice',
           'choices',
           'expovariate',
           'gammavariate',
           'gauss',
           'getrandbits',
           'getstate',
           'lognormvariate',
           'normalvariate',
           'paretovariate',
           'randbytes',
           'randint',
           'random',
           'randrange',
           'sample',
           'seed',
 'setstate',
 'shuffle',
 'triangular',
 'uniform',
 'vonmisesvariate',
 'weibullvariate']
```

In [22]:
```
help(random.randint)
```

```
Help on method randint in module random:

randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.
```

In [23]:
```
random.randint(1,20)
```

Out[23]:
```
12
```

In [24]:
```
# package name:  random
# method name:  random
import random
dir(random)
help(random.random)
```

```
Help on built-in function random:

random() method of random.Random instance
    random() -> x in the interval [0, 1).
```

In [25]:
```
random.random()
```

Out[25]:
```
0.1694746898871523
```

math

package name : math

method pi

sqrt

sin

In [26]:
```
import math
```

In [27]:
```
dir(math)
```

```
Out[27]: ['__doc__',
          '__loader__',
          '__name__',
          '__package__',
          '__spec__',
          'acos',
          'acosh',
          'asin',
          'asinh',
          'atan',
          'atan2',
          'atanh',
          'cbrt',
          'ceil',
          'comb',
          'copysign',
          'cos',
          'cosh',
          'degrees',
          'dist',
          'e',
          'erf',
          'erfc',
          'exp',
          'exp2',
          'expm1',
          'fabs',
          'factorial',
          'floor',
          'fmod',
          'frexp',
          'fsum',
          'gamma',
          'gcd',
          'hypot',
          'inf',
          'isclose',
          'isfinite',
          'isinf',
          'isnan',
          'isqrt',
          'lcm',
          'ldexp',
          'lgamma',
          'log',
          'log10',
          'log1p',
          'log2',
          'modf',
          'nan',
          'nextafter',
          'perm',
          'pi',
          'pow',
          'prod',
          'radians',
          'remainder',
          'sin',
          'sinh',
          'sqrt',
          'tan',
          'tanh',
          'tau',
          'trunc',
          'ulp']
```

In [28]:
```
# pi
# pow
# sqrt
# sin

help(math.sqrt)
```

```
Help on built-in function sqrt in module math:

sqrt(x, /)
    Return the square root of x.
```

In [29]:
```
math.sqrt(25)
```

Out[29]:
```
5.0
```

In [30]:
```
help(math.pow)
```

```
Help on built-in function pow in module math:

pow(x, y, /)
    Return x**y (x to the power of y).
```

In [31]:
```
math.pow(2,3)
```

Out[31]:
```
8.0
```

In [32]:
```
help(math.sin)
```

```
Help on built-in function sin in module math:

sin(x, /)
    Return the sine of x (measured in radians).
```

In [33]:
```
math.sin(90)
```

Out[33]:
```
0.8939966636005579
```

In [ ]:
```
math.sqrt(25)
math.pow(2,3)
math.sin(90)
```

In [34]:
```
math.sqrt(x=25)
# curosr inside the bracker
# then apply shift+tab

# / is mentioned means
# do not provide x=25 values
# directly give 25
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[34], line 1
----> 1 math.sqrt(x=25)

TypeError: math.sqrt() takes no keyword arguments
```

```
In [35]: math.sqrt
```

```
Out[35]: <function math.sqrt(x, /)>
```

- function means we are forgetting brackets
- we need to keep brackets
- bound method also means forgetting the brackets only
- not callable means remove bracket
- whenever you see slash don't provide variable name
- directly give the value

```
In [38]: math.sqrt(25)
```

```
Out[38]: 5.0
```

```
In [42]: random.randint(10,20)
```

```
Out[42]: 13
```

```
In [47]: random.random()
```

```
Out[47]: 0.12403115023466815
```

```
In [44]: dir(random)
```

```
Out[44]: ['BPF',
          'LOG4',
          'NV_MAGICCONST',
          'RECIP_BPF',
          'Random',
          'SG_MAGICCONST',
          'SystemRandom',
          'TWOPI',
          '_ONE',
          '_Sequence',
          '_Set',
          '__all__',
          '__builtins__',
          '__cached__',
          '__doc__',
          '__file__',
          '__loader__',
          '__name__',
          '__package__',
          '__spec__',
          '_accumulate',
          '_acos',
          '_bisect',
          '_ceil',
          '_cos',
          '_e',
          '_exp',
          '_floor',
          '_index',
          '_inst',
          '_isfinite',
          '_log',
          '_os',
          '_pi',
          '_random',
          '_repeat',
          '_sha512',
          '_sin',
          '_sqrt',
          '_test',
          '_test_generator',
          '_urandom',
          '_warn',
          'betavariate',
          'choice',
          'choices',
          'expovariate',
          'gammavariate',
          'gauss',
          'getrandbits',
          'getstate',
          'lognormvariate',
          'normalvariate',
          'paretovariate',
          'randbytes',
          'randint',
          'random',
          'randrange',
          'sample',
          'seed',
 'setstate',
 'shuffle',
 'triangular',
 'uniform',
 'vonmisesvariate',
 'weibullvariate']
```

In [49]:
```
math.pi
# direct values never include brackets
```

Out[49]:
```
3.141592653589793
```

In [50]:
```
# package name: keyword
# method name: kwlist
import keyword
```

In [52]:
```
len(keyword.kwlist)
```

Out[52]:
```
35
```

In [ ]:
```
# step-1:
# import <package_name>

# step-2:
# dir(<package_name>)
#    methods will display

# step-3:
# help(<package_name>.<method_name>)

# step-4:
# <package_name>.<method_name>()  # 99%
```

In [1]:
```
import random
```

In [2]:
```
dir(random)
```

```
Out[2]: ['BPF',
         'LOG4',
         'NV_MAGICCONST',
         'RECIP_BPF',
         'Random',
         'SG_MAGICCONST',
         'SystemRandom',
         'TWOPI',
         '_ONE',
         '_Sequence',
         '_Set',
         '__all__',
         '__builtins__',
         '__cached__',
         '__doc__',
         '__file__',
         '__loader__',
         '__name__',
         '__package__',
         '__spec__',
         '_accumulate',
         '_acos',
         '_bisect',
         '_ceil',
         '_cos',
         '_e',
         '_exp',
         '_floor',
         '_index',
         '_inst',
         '_isfinite',
         '_log',
         '_os',
         '_pi',
         '_random',
         '_repeat',
         '_sha512',
         '_sin',
         '_sqrt',
         '_test',
         '_test_generator',
         '_urandom',
         '_warn',
         'betavariate',
         'choice',
         'choices',
         'expovariate',
         'gammavariate',
         'gauss',
         'getrandbits',
         'getstate',
         'lognormvariate',
         'normalvariate',
         'paretovariate',
         'randbytes',
         'randint',
         'random',
         'randrange',
         'sample',
         'seed',
 'setstate',
 'shuffle',
 'triangular',
 'uniform',
 'vonmisesvariate',
 'weibullvariate']
```

In [3]:
```
help(random.randint)
```

```
Help on method randint in module random:

randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.
```

In [5]:
```
random.randint(10,20)
```

Out[5]:
```
13
```

In [6]:
```
import math
```

In [8]:
```
math.pi
```

Out[8]:
```
3.141592653589793
```

() : function or methods

only functions or methods callable

if something says not callable means , it is not a function

it is not a function means, you need to remove bracket

bound method means ==== add the brackets

function ======= add the brackets

not callable ==== remove the brackets

- random
- math
- keyword

time

In [12]:
```
import time

print("Father:hello")
time.sleep(2)
print("D: hai papa how are you")
time.sleep(2)
print("Father:do you have school today")
```

```
Father:hello
D: hai papa how are you
Father:do you have school today
```

In [ ]:
```
Computer vision is used to image operations
video operations

we called it as opencv

pcakge name: cv2
```

In [15]:
```
import cv2
```

- whenever module not fund we need to install it
- here our package name is cv2
- 99% the installation will be like this

pip install

- some packages installation name will different and python import name is different
- in order to install the packages we need internet
- if internet problem occures while installing we will get http error

In [16]:
```
import streamlit
```

Note

pip freeze is the command to know the aleady existed packages in our laptop

In [17]:
```
import numpy
```

In [18]:
```
numpy
```

Out[18]:
```
<module 'numpy' from 'C:\\Users\\omkar\\anaconda3\\Lib\\site-packages\\numpy\\_
_init__.py'>
```

!pip install opencv-python

- in jupyter notebook we need add ! mark
- in anaconda prompt no need of ! mark
- directly name you can provide

In [19]:
```
# hard coding: static
# we are fixing the values
number1=10
number2=20
add=number1+number2
print(add)
```

```
30
```

In [20]:
```
# dynamic
number1=eval(input("enter the number 1:"))
number2=eval(input("enter the number 2:"))
add=number1+number2
print(add)
```

```
300
```

In [23]:
```
# we can take the numbers randomly also
import random
number1=random.randint(1,100)
number2=random.randint(100,200)
add=number1+number2
print(f"the addition of {number1} and {number2} is {add}")
```

```
the addition of 25 and 125 is 150
```

How we pass the numbers

- hard codings
- using keyboard
- using random package , taking numbers randomly

In [ ]:
```
############################################################
number1=10
number2=20
add=number1+number2
print(add)

#############################################################
number1=eval(input("enter the number 1:"))
number2=eval(input("enter the number 2:"))
add=number1+number2
print(add)

################################################################
import random
number1=random.randint(1,100)
number2=random.randint(100,200)
add=number1+number2
print(f"the addition of {number1} and {number2} is {add}")
```

In [26]:
```
n1=10
n2=eval(input("enter the n2:"))
n3=random.randint(1,100)
avg=(n1+n2+n3)/3
print(f"The avergae of {n1},{n2} and {n3} is: {avg}")
```

```
The avergae of 10,10 and 48 is: 22.666666666666668
```

In [ ]:
```
# in the entire notebook
# if you import packages
# no need to import every time
# just verify it the line is executed or not
# all the packages we will import at starting only line
```

```
In [ ]: # assignment-2
        # assignment-1 qns only but
        # you need to take random values every

        # 11 the print statement using time.sleep
```

round

```
In [27]: avg=22.666666666666668
         round(avg)

Out[27]: 23
```

```
In [29]: avg=22.666666666666668
         round(avg,3)

Out[29]: 22.667
```

```
In [ ]: type()
        input()
        print()
        round()
```

```
In [ ]: n1=10
        n2=eval(input("enter the n2:"))
        n3=random.randint(1,100)
        avg=(n1+n2+n3)/3
        avg1=round(avg,2)
        print(f"The avergae of {n1},{n2} and {n3} is: {avg1}")
```

```
In [ ]: n1=10
        n2=eval(input("enter the n2:"))
        n3=random.randint(1,100)
        avg=round((n1+n2+n3)/3,2)

        print(f"The avergae of {n1},{n2} and {n3} is: {avg1}")
```