In [ ]:
```
package name: random
              math
              time
              cv2
```

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

In [2]:
```
import math
```

In [3]:
```
import time
```

In [4]:
```
import cv2
```

In [ ]:
```
dir(<package name>)
```

In [5]:
```
import random
```

In [6]:
```
dir(random)
```

Out[6]:
```
['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 [ ]:
```
help(<packagename>.<method name>)
package name: random
method name : randint
```

In [7]:
```
import random
dir(random)
```

Out[7]:
```
['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 [8]:
```
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 [12]:
```
random.randint(1,10)
```

Out[12]: 6

- Step1: import
  - ex: import random
- Step-2: dir()
  - there are so many methods are there in random
  - dir(random)
- Step-3: help(.)
  - suppose I want to use randint method
  - help(random.randint)
- Step-4: apply the code
  - I understood what randint will do
  - random.randint(1,10)

In [13]:
```
import random
dir(random)
```

Out[13]:
```
['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 [14]:
```
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 [18]:
```
random.randint(10,20)
```

Out[18]: 17

random

In [19]:
```
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]: 0.6383610880186024

In [21]:
```
random.randint(1,10)
```

Out[21]: 7

In [23]:
```
random.random(10)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[23], line 1
----> 1 random.random(10)

TypeError: Random.random() takes no arguments (1 given)
```

In [ ]:
```
#packagename: keyword
```

In [26]:
```
import keyword
```

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

Out[27]:
```
['__all__',
 '__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 'iskeyword',
 'issoftkeyword',
 'kwlist',
 'softkwlist']
```

In [28]:
```
help(keyword.kwlist)

# use of the method
```

```
Help on list object:

class list(object)
 |  list(iterable=(), /)
 |
 |  Built-in mutable sequence.
 |
 |  If no argument is given, the constructor creates a new empty list.
 |  The argument must be an iterable if specified.
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return key in self.
 |
 |  __delitem__(self, key, /)
 |      Delete self[key].
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(...)
 |      x.__getitem__(y) <==> x[y]
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __iadd__(self, value, /)
 |      Implement self+=value.
 |
 |  __imul__(self, value, /)
 |      Implement self*=value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.
 |
```

```
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __reversed__(self, /)
 |      Return a reverse iterator over the list.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(self, /)
 |      Return the size of the list in memory, in bytes.
 |
 |  append(self, object, /)
 |      Append object to the end of the list.
 |
 |  clear(self, /)
 |      Remove all items from list.
 |
 |  copy(self, /)
 |      Return a shallow copy of the list.
 |
 |  count(self, value, /)
 |      Return number of occurrences of value.
 |
 |  extend(self, iterable, /)
 |      Extend list by appending elements from the iterable.
 |
 |  index(self, value, start=0, stop=9223372036854775807, /)
 |      Return first index of value.
 |
 |      Raises ValueError if the value is not present.
 |
 |  insert(self, index, object, /)
 |      Insert object before index.
 |
 |  pop(self, index=-1, /)
 |      Remove and return item at index (default last).
 |
 |      Raises IndexError if list is empty or index is out of range.
 |
 |  remove(self, value, /)
 |      Remove first occurrence of value.
 |
 |      Raises ValueError if the value is not present.
 |
 |  reverse(self, /)
 |      Reverse *IN PLACE*.
 |
 |  sort(self, /, *, key=None, reverse=False)
 |      Sort the list in ascending order and return None.
 |
 |      The sort is in-place (i.e. the list itself is modified) and stable (i.e.
the
 |      order of two equal elements is maintained).
```

```
 |
 |      If a key function is given, apply it once to each list item and sort the
m,
 |      ascending or descending, according to their function values.
 |
 |      The reverse flag can be set to sort in descending order.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...) from builtins.type
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None
```

In [29]:
```
keyword.kwlist
```

Out[29]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 'break',
 'class',
 'continue',
 'def',
 'del',
 'elif',
 'else',
 'except',
 'finally',
 'for',
 'from',
 'global',
 'if',
 'import',
 'in',
 'is',
 'lambda',
 'nonlocal',
 'not',
 'or',
 'pass',
 'raise',
 'return',
 'try',
 'while',
 'with',
 'yield']
```

In [30]:
```
len(keyword.kwlist)
```

Out[30]:
```
35
```

In [ ]:
```
why we did not use () for kwlist as randint()
```

math

In [31]:
```
import math
```

In [32]:
```
dir(math)
```

Out[32]:
```
['__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 [ ]:
```
- pi

- sin

- sqrt

- pow
```

In [34]:
```
help(math.pi)
```

```
Help on float object:

class float(object)
 |  float(x=0, /)
 |
 |  Convert a string or number to a floating point number, if possible.
 |
 |  Methods defined here:
 |
 |  __abs__(self, /)
 |      abs(self)
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __bool__(self, /)
 |      True if self else False
 |
 |  __ceil__(self, /)
 |      Return the ceiling as an Integral.
 |
 |  __divmod__(self, value, /)
 |      Return divmod(self, value).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __float__(self, /)
 |      float(self)
 |
 |  __floor__(self, /)
 |      Return the floor as an Integral.
 |
 |  __floordiv__(self, value, /)
 |      Return self//value.
 |
 |  __format__(self, format_spec, /)
 |      Formats the float according to format_spec.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getnewargs__(self, /)
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __int__(self, /)
 |      int(self)
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __lt__(self, value, /)
```

```
|      Return self<value.
|
|  __mod__(self, value, /)
|      Return self%value.
|
|  __mul__(self, value, /)
|      Return self*value.
|
|  __ne__(self, value, /)
|      Return self!=value.
|
|  __neg__(self, /)
|      -self
|
|  __pos__(self, /)
|      +self
|
|  __pow__(self, value, mod=None, /)
|      Return pow(self, value, mod).
|
|  __radd__(self, value, /)
|      Return value+self.
|
|  __rdivmod__(self, value, /)
|      Return divmod(value, self).
|
|  __repr__(self, /)
|      Return repr(self).
|
|  __rfloordiv__(self, value, /)
|      Return value//self.
|
|  __rmod__(self, value, /)
|      Return value%self.
|
|  __rmul__(self, value, /)
|      Return value*self.
|
|  __round__(self, ndigits=None, /)
|      Return the Integral closest to x, rounding half toward even.
|
|      When an argument is passed, work like built-in round(x, ndigits).
|
|  __rpow__(self, value, mod=None, /)
|      Return pow(value, self, mod).
|
|  __rsub__(self, value, /)
|      Return value-self.
|
|  __rtruediv__(self, value, /)
|      Return value/self.
|
|  __sub__(self, value, /)
|      Return self-value.
|
|  __truediv__(self, value, /)
|      Return self/value.
|
|  __trunc__(self, /)
|      Return the Integral closest to x between 0 and x.
```

```
 |
 |  as_integer_ratio(self, /)
 |      Return integer ratio.
 |
 |      Return a pair of integers, whose ratio is exactly equal to the original f
loat
 |      and with a positive denominator.
 |
 |      Raise OverflowError on infinities and a ValueError on NaNs.
 |
 |      >>> (10.0).as_integer_ratio()
 |      (10, 1)
 |      >>> (0.0).as_integer_ratio()
 |      (0, 1)
 |      >>> (-.25).as_integer_ratio()
 |      (-1, 4)
 |
 |  conjugate(self, /)
 |      Return self, the complex conjugate of any float.
 |
 |  hex(self, /)
 |      Return a hexadecimal representation of a floating-point number.
 |
 |      >>> (-0.1).hex()
 |      '-0x1.999999999999ap-4'
 |      >>> 3.14159.hex()
 |      '0x1.921f9f01b866ep+1'
 |
 |  is_integer(self, /)
 |      Return True if the float is an integer.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __getformat__(typestr, /) from builtins.type
 |      You probably don't want to use this function.
 |
 |        typestr
 |          Must be 'double' or 'float'.
 |
 |      It exists mainly to be used in Python's test suite.
 |
 |      This function returns whichever of 'unknown', 'IEEE, big-endian' or 'IEE
E,
 |      little-endian' best describes the format of floating point numbers used b
y the
 |      C type named by typestr.
 |
 |  fromhex(string, /) from builtins.type
 |      Create a floating-point number from a hexadecimal string.
 |
 |      >>> float.fromhex('0x1.ffffp10')
 |      2047.984375
 |      >>> float.fromhex('-0x1p-1074')
 |      -5e-324
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs) from builtins.type
```

```
|      Create and return a new object.  See help(type) for accurate signature.
|
|  ----------------------------------------------------------------------
|  Data descriptors defined here:
|
|  imag
|      the imaginary part of a complex number
|
|  real
|      the real part of a complex number
```

In [35]:
```
math.pi   # pi= 22/7=3.14
```

Out[35]: 3.141592653589793

In [36]:
```
help(math.sqrt)
```

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

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

In [41]:
```
math.sqrt(25)

# function means i forgot brackets
```

Out[41]: 5.0

In [44]:
```
math.sin(90)
```

Out[44]: 0.8939966636005579

In [45]:
```
help(math.sin)
```

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

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

In [46]:
```
math.sin(45)
# Move the cursor inside brackets
# apply  shift+tab
```

Out[46]: 0.8509035245341184

In [50]:
```
math.pow(2,5)
```

Out[50]: 32.0

In [ ]:
```
math.pi
math.sin(90)
math.sqrt(25)
math.pow(2,5)
```

In [51]:
```
math.sqrt
```

Out[51]:
```
<function math.sqrt(x, /)>
```

In [53]:
```
math.sin()
```

Out[53]:
```
<function math.sin(x, /)>
```

In [54]:
```
help(math.sin)
```

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

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

In [55]:
```
math.sin
```

Out[55]:
```
<function math.sin(x, /)>
```

In [56]:
```
import time
```

In [57]:
```
time.sleep(5)
print(10)
```

```
10
```

In [59]:
```
import random
import math
import time
num=random.randint(1,10)
num2=math.sqrt(100)
time.sleep(5)
print(num)
time.sleep(5)
print(num2)
```

```
5
10.0
```

- whenever you got the error first understand the error
- if package is not available:

  No module name : <package name>

- 90 percent the syntax is

  pip install <package name>

- If the package name is correct or not we can check in google
- But you need to confirm with python organization

In [60]:
```
import cv2
```

In [61]:
```
import streamlit
```

```
# get the error
# pip install streamlit
```

In [64]:
```
math.pi
```

Out[64]:
```
3.141592653589793
```

In [ ]:
```

```