```
In [1]: 1 + 1 # Addition
Out[1]: 2

In [3]: 1-1 # Substraction
Out[3]: 0

In [7]: 3 * 4 # Multiplication
Out[7]: 12

In [866… 8 / 4 # float division
Out[866… 2.0

In [868… 8 //5 # floor or integer division
Out[868… 1

In [870… 2 + (5 * 6) -3 # BODMAS
Out[870… 29

In [872… 2 * 2 * 2 * 2 # exponential
Out[872… 16

In [874… 15 % 2
Out[874… 1

In [876… a,b,c,d,e = 15, 3.14,'nit', 8+ 5j, True

         print(a)
         print(b)
         print(c)
         print(d)
         print(e)

15
3.14
nit
(8+5j)
True

In [878… 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 [880… 'Naresh IT'
Out[880… 'Naresh IT'

In [882… print('Naresh IT')
Naresh IT

In [884… print('It's interesting learning data science at naresh IT from Prakash Senepathi Sir')

  Cell In[884], line 1
    print('It's interesting learning data science at naresh IT from Prakash Senepathi Sir')
                                                                                          ^
SyntaxError: unterminated string literal (detected at line 1)

In [885… print('It\'s interesting learning data science at naresh IT from Prakash Senepathi Sir') #\ has to ignore the above error
It's interesting learning data science at naresh IT from Prakash Senepathi Sir

In [888… print('Naresh IT', 'technology')
Naresh IT technology

In [889… # print the nit 2 times
         'nit' + 'nit'
Out[889… 'nitnit'

In [892… 'nit'  'nit'
Out[892… 'nitnit'

In [893… 5 * 'nit ' # printing 5 times
Out[893… 'nit nit nit nit nit '

In [896… print('C:\nit') #\n -- new line
C:
it
```

```
In [897… print(r'C:\nit') # raw string
```

```
C:\nit
```

```
In [898… x = 4 #x is a variable/object/identifier
         x
```

```
Out[898… 4
```

```
In [902… x + 3
```

```
Out[902… 7
```

```
In [903… y =12
         y
```

```
Out[903… 12
```

```
In [904… x + y
```

```
Out[904… 16
```

```
In [905… _+ y # _understand the previous result
```

```
Out[905… 17
```

```
In [909… y
```

```
Out[909… 12
```

```
In [910… result = _ + y
         result
```

```
Out[910… 17
```

```
In [914… _ = 5
         y = 6

         result = _ + y
         result
```

```
Out[914… 11
```

```
In [915… name ='mit'
         name
```

```
Out[915… 'mit'
```

```
In [918… name = name + 'technology'
```

```
In [920… len(name)
```

```
Out[920… 13
```

```
In [922… name[0] # in python index begins with 0
```

```
Out[922… 'm'
```

```
In [924… name[5]
```

```
Out[924… 'c'
```

```
In [926… name[-1]
```

```
Out[926… 'y'
```

Slicing

```
In [929… name
```

```
Out[929… 'mittechnology'
```

```
In [931… name[0:2]
```

```
Out[931… 'mi'
```

```
In [932… name[:4]
```

```
Out[932… 'mitt'
```

```
In [935… name[13:2:2]
```

```
Out[935… ''
```

```
In [937… name[13:4:5]
```

```
Out[937… ''
```

```
In [939… name
```

```
Out[939… 'mittechnology'
```

```
In [940… name[14]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[940], line 1
----> 1 name[14]

IndexError: string index out of range
```

In [ ]:
```
name1='fine'
name1
```

In [ ]:
```
name[0:2]
```

In [ ]:
```
name[1:]
```

List

In [944…
```
l =[]
```

In [948…
```
nums = [10,20,30]
```

In [950…
```
nums
```

Out[950…
```
[10, 20, 30]
```

In [952…
```
nums[2]
```

Out[952…
```
30
```

In [954…
```
#Nested List
nums2 = ['hi', 23, 3.14, True, nums]
nums2
```

Out[954…
```
['hi', 23, 3.14, True, [10, 20, 30]]
```

In [955…
```
nums2.append(45) # append a value to an existing list
nums
```

Out[955…
```
[10, 20, 30]
```

In [956…
```
nums.remove(0,2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[956], line 1
----> 1 nums.remove(0,2)

TypeError: list.remove() takes exactly one argument (2 given)
```

In [957…
```
nums.remove(1)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[957], line 1
----> 1 nums.remove(1)

ValueError: list.remove(x): x not in list
```

In [958…
```
nums.remove(nums[1])
```

In [961…
```
nums
```

Out[961…
```
[10, 30]
```

In [963…
```
nums2
```

Out[963…
```
['hi', 23, 3.14, True, [10, 30], 45]
```

In [965…
```
nums.pop(1)
nums
```

Out[965…
```
[10]
```

In [967…
```
nums2
```

Out[967…
```
['hi', 23, 3.14, True, [10], 45]
```

In [969…
```
nums2.remove()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[969], line 1
----> 1 nums2.remove()

TypeError: list.remove() takes exactly one argument (0 given)
```

In [ ]:
```
nums2
```

In [972…
```
nums2.append('Vinay')
nums2
```

Out[972…
```
['hi', 23, 3.14, True, [10], 45, 'Vinay']
```

In [974…
```
nums.pop() #if we don't pass agruments when it will consider last index
```

Out[974…
```
10
```

```
In [976… nums

Out[976… []

In [978… nums.append(23)
         nums.append(45)
         nums

Out[978… [23, 45]

In [979… nums2

Out[979… ['hi', 23, 3.14, True, [23, 45], 45, 'Vinay']

In [980… nums2.pop()
         nums2

Out[980… ['hi', 23, 3.14, True, [23, 45], 45]

In [984… nums2.insert(0,1)
         nums2

Out[984… [1, 'hi', 23, 3.14, True, [23, 45], 45]

In [985… del nums2[2:]
         nums2

Out[985… [1, 'hi']

In [988… nums2.extend([29,14,35])
         nums2

Out[988… [1, 'hi', 29, 14, 35]
```

min(num2)

```
In [991… min(nums2)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[991], line 1
----> 1 min(nums2)

TypeError: '<' not supported between instances of 'str' and 'int'

In [993… max(nums2)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[993], line 1
----> 1 max(nums2)

TypeError: '>' not supported between instances of 'str' and 'int'

In [995… sum(nums2)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[995], line 1
----> 1 sum(nums2)

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

In [997… num123 = [1,2,4,6,5,7]
         num123

Out[997… [1, 2, 4, 6, 5, 7]

In [999… min(num123)

Out[999… 1

In [100… max(num123)

Out[100… 7

In [100… sum(num123)

Out[100… 25

In [100… num123.sort()

In [100… num123

Out[100… [1, 2, 4, 5, 6, 7]

In [100… num123.sort(desc)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1009], line 1
----> 1 num123.sort(desc)

NameError: name 'desc' is not defined

In [ ]: num123.sort(0)
In [101…  l =[1,2,3]
          l

Out[101…  [1, 2, 3]

In [101…  l[0]

Out[101…  1
```

Tuple

```
In [101…  # TUPLE
          tup = (15,25,36)
          tup

Out[101…  (15, 25, 36)

In [101…  tup[0]

Out[101…  15

In [102…  tup[0] =20

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1021], line 1
----> 1 tup[0] =20

TypeError: 'tuple' object does not support item assignment

In [ ]:   tup[1]

In [102…  tup

Out[102…  (15, 25, 36)

In [102…  tip1 =(1,' test', True)

In [102…  tip1

Out[102…  (1, ' test', True)

In [102…  tip1[2]

Out[102…  True

In [103…  tip1

Out[103…  (1, ' test', True)

In [103…  tip1.pop()

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[1034], line 1
----> 1 tip1.pop()

AttributeError: 'tuple' object has no attribute 'pop'

In [ ]:   tup.count

In [103…  tup.count()

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1035], line 1
----> 1 tup.count()

TypeError: tuple.count() takes exactly one argument (0 given)

In [ ]:   a = tup.count
          a

In [104…  a

Out[104…  15

In [104…  len(tup)

Out[104…  3
```

SET

```
In [104…  S ={}

In [104…  S

Out[104…  {}

In [104…  S1 = {1,23,5.1,2.34}
          S1

Out[104…  {1, 2.34, 5.1, 23}
```

```
In [104… S1.clear()

In [104… S1

Out[104… set()

In [104… S1.add(1)
         S1.add(3.14)
         S1.add('nit')
         S1.add(True)
         S1

Out[104… {1, 3.14, 'nit'}

In [105… S1.add(True)

In [105… S1

Out[105… {1, 3.14, 'nit'}

In [105… S1.update(0) =3

  Cell In[1052], line 1
    S1.update(0) =3
    ^
SyntaxError: cannot assign to function call here. Maybe you meant '==' instead of '='?

In [105… S1

Out[105… {1, 3.14, 'nit'}

In [105… S1.add(True)

In [105… len(S1)

Out[105… 3

In [105… S1.add(3)

In [105… len(S1)

Out[105… 4

In [105… s2 = {True}
         s2

Out[105… {True}

In [105… s2.add(True)

In [106… s2

Out[106… {True}

In [107… s2.remove(0)

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[1079], line 1
----> 1 s2.remove(0)

KeyError: 0

In [ ]: s2.pop(0)

In [108… s2.pop()
         s2

Out[108… set()

In [108… S1

Out[108… {1, 3, 3.14, 'nit'}

In [108… s1[1]

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1086], line 1
----> 1 s1[1]

NameError: name 's1' is not defined

In [108… S1[1]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1088], line 1
----> 1 S1[1]

TypeError: 'set' object is not subscriptable

In [108… S1[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1089], line 1
----> 1 S1[0]

TypeError: 'set' object is not subscriptable
```

```
In [109…  S1(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1092], line 1
----> 1 S1(0)

TypeError: 'set' object is not callable
```

```
In [ ]:  S1[0]
```

```
In [109…  S1
```

```
Out[109…  {1, 3, 3.14, 'nit'}
```

```
In [109…  len(s1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1096], line 1
----> 1 len(s1)

NameError: name 's1' is not defined
```

```
In [109…  len(S1)
```

```
Out[109…  4
```

DICTIONARY

```
In [110…  # DICTIONARY
          data = {1:'apple', 2:'banana', 4:'orange'}
          data
```

```
Out[110…  {1: 'apple', 2: 'banana', 4: 'orange'}
```

```
In [110…  list1 =[34]
          list1
```

```
Out[110…  [34]
```

```
In [110…  data.items
```

```
Out[110…  <function dict.items>
```

```
In [110…  data.values
```

```
Out[110…  <function dict.values>
```

```
In [110…  data
```

```
Out[110…  {1: 'apple', 2: 'banana', 4: 'orange'}
```

```
In [111…  data[3] = 3.14
```

```
In [111…  data
```

```
Out[111…  {1: 'apple', 2: 'banana', 4: 'orange', 3: 3.14}
```

```
In [111…  data.sort()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[1115], line 1
----> 1 data.sort()

AttributeError: 'dict' object has no attribute 'sort'
```

```
In [111…  data.popitem(4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1116], line 1
----> 1 data.popitem(4)

TypeError: dict.popitem() takes no arguments (1 given)
```

```
In [111…  data.pop(4)
```

```
Out[111…  'orange'
```

```
In [112…  data
```

```
Out[112…  {1: 'apple', 2: 'banana', 3: 3.14}
```

```
In [112…  data[3]
```

```
Out[112…  3.14
```

In [112…

```
help()
```

```
Welcome to Python 3.12's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.12/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q" or "quit".

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
```

In [663…

```
help(list)
```

```
Help on class list in module builtins:

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 bool(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__(self, index, /)
 |      Return self[index].
 |
 |  __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 them,
 |      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__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None
```

In [665…

```
builtins
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[665], line 1
----> 1 builtins

NameError: name 'builtins' is not defined
```

In [667…

```
help()
```

```
Welcome to Python 3.12's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.12/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q" or "quit".
```

```
Help on built-in module builtins:

NAME
    builtins - Built-in functions, types, exceptions, and other objects.

DESCRIPTION
    This module provides direct access to all 'built-in'
    identifiers of Python; for example, builtins.len is
    the full name for the built-in function len().

    This module is not normally accessed explicitly by most
    applications, but can be useful in modules that provide
    objects with the same name as a built-in value, but in
    which the built-in of that name is also needed.

CLASSES
    object
        BaseException
            BaseExceptionGroup
                ExceptionGroup(BaseExceptionGroup, Exception)
            Exception
                ArithmeticError
                    FloatingPointError
                    OverflowError
                    ZeroDivisionError
                AssertionError
                AttributeError
                BufferError
                EOFError
                ImportError
                    ModuleNotFoundError
                LookupError
                    IndexError
                    KeyError
                MemoryError
                NameError
                    UnboundLocalError
                OSError
                    BlockingIOError
                    ChildProcessError
                    ConnectionError
                        BrokenPipeError
                        ConnectionAbortedError
                        ConnectionRefusedError
                        ConnectionResetError
                    FileExistsError
                    FileNotFoundError
                    InterruptedError
                    IsADirectoryError
                    NotADirectoryError
                    PermissionError
                    ProcessLookupError
                    TimeoutError
                ReferenceError
                RuntimeError
                    NotImplementedError
                    RecursionError
                StopAsyncIteration
                StopIteration
                SyntaxError
                    IndentationError
                        TabError
                SystemError
                TypeError
                ValueError
                    UnicodeError
                        UnicodeDecodeError
                        UnicodeEncodeError
                        UnicodeTranslateError
                Warning
                    BytesWarning
                    DeprecationWarning
                    EncodingWarning
                    FutureWarning
                    ImportWarning
                    PendingDeprecationWarning
                    ResourceWarning
                    RuntimeWarning
                    SyntaxWarning
                    UnicodeWarning
                    UserWarning
            GeneratorExit
            KeyboardInterrupt
            SystemExit
        bytearray
        bytes
        classmethod
        complex
        dict
        enumerate
        filter
        float
        frozenset
        int
            bool
        list
        map
        memoryview
        property
        range
        reversed
        set
        slice
        staticmethod
        str
        super
        tuple
        type
        zip

    class ArithmeticError(Exception)
     |  Base class for arithmetic errors.
     |
     |  Method resolution order:
     |      ArithmeticError
     |      Exception
     |      BaseException
     |      object
     |
     |  Built-in subclasses:
     |      FloatingPointError
     |      OverflowError
     |      ZeroDivisionError
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class AssertionError(Exception)
     |  Assertion failed.
     |
     |  Method resolution order:
     |      AssertionError
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
```

```
 |      Helper for pickle.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args

class AttributeError(Exception)
 |  Attribute not found.
 |  
 |  Method resolution order:
 |      AttributeError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __getstate__(...)
 |      Helper for pickle.
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |  
 |  name
 |      attribute name
 |  
 |  obj
 |      object
 |  
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |  
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class BaseException(object)
 |  Common base class for all exceptions
 |
 |  Built-in subclasses:
 |      BaseExceptionGroup
 |      Exception
 |      GeneratorExit
 |      KeyboardInterrupt
 |      ... and 1 other subclasses
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class BaseExceptionGroup(BaseException)
 |  A combination of multiple unrelated exceptions.
 |
 |  Method resolution order:
 |      BaseExceptionGroup
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      ExceptionGroup
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  derive(...)
 |
 |  split(...)
 |
 |  subgroup(...)
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  exceptions
 |      nested exceptions
 |
 |  message
 |      exception message
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class BlockingIOError(OSError)
 |  I/O operation would block.
 |
 |  Method resolution order:
 |      BlockingIOError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
```

```
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class BrokenPipeError(ConnectionError)
 |  Broken pipe.
 |
 |  Method resolution order:
 |      BrokenPipeError
 |      ConnectionError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
```

```
     |      set self.__traceback__ to tb and return self.
     |  
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |  
     |  __cause__
     |      exception cause
     |  
     |  __context__
     |      exception context
     |  
     |  __dict__
     |  
     |  __suppress_context__
     |  
     |  __traceback__
     |  
     |  args
    
    class BufferError(Exception)
     |  Buffer error.
     |  
     |  Method resolution order:
     |      BufferError
     |      Exception
     |      BaseException
     |      object
     |  
     |  Methods defined here:
     |  
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |  
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |  
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |  
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |  
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |  
     |  __reduce__(...)
     |      Helper for pickle.
     |  
     |  __repr__(self, /)
     |      Return repr(self).
     |  
     |  __setstate__(...)
     |  
     |  __str__(self, /)
     |      Return str(self).
     |  
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |  
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |  
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |  
     |  __cause__
     |      exception cause
     |  
     |  __context__
     |      exception context
     |  
     |  __dict__
     |  
     |  __suppress_context__
     |  
     |  __traceback__
     |  
     |  args
    
    class BytesWarning(Warning)
     |  Base class for warnings about bytes and buffer related problems, mostly
     |  related to conversion from str or comparing to str.
     |  
     |  Method resolution order:
     |      BytesWarning
     |      Warning
     |      Exception
     |      BaseException
     |      object
     |  
     |  Methods defined here:
     |  
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |  
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |  
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
```

```
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ChildProcessError(OSError)
 |  Child process error.
 |
 |  Method resolution order:
 |      ChildProcessError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ConnectionAbortedError(ConnectionError)
 |  Connection aborted.
 |
 |  Method resolution order:
 |      ConnectionAbortedError
 |      ConnectionError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
```

```
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class ConnectionError(OSError)
     |  Connection error.
     |
     |  Method resolution order:
     |      ConnectionError
     |      OSError
     |      Exception
     |      BaseException
     |      object
     |
     |  Built-in subclasses:
     |      BrokenPipeError
     |      ConnectionAbortedError
     |      ConnectionRefusedError
     |      ConnectionResetError
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from OSError:
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  ----------------------------------------------------------------------
     |  Static methods inherited from OSError:
     |
     |  __new__(*args, **kwargs) class method of OSError
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from OSError:
     |
     |  characters_written
     |
     |  errno
     |      POSIX exception code
     |
     |  filename
     |      exception filename
     |
     |  filename2
     |      second exception filename
     |
     |  strerror
     |      exception strerror
     |
     |  winerror
     |      Win32 exception code
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args
    
    class ConnectionRefusedError(ConnectionError)
     |  Connection refused.
     |  
     |  Method resolution order:
     |      ConnectionRefusedError
     |      ConnectionError
     |      OSError
     |      Exception
     |      BaseException
     |      object
     |  
     |  Methods defined here:
     |  
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |  
     |  ----------------------------------------------------------------------
     |  Methods inherited from OSError:
     |  
     |  __reduce__(...)
     |      Helper for pickle.
     |  
     |  __str__(self, /)
     |      Return str(self).
     |  
     |  ----------------------------------------------------------------------
     |  Static methods inherited from OSError:
     |  
     |  __new__(*args, **kwargs) class method of OSError
     |      Create and return a new object.  See help(type) for accurate signature.
     |  
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from OSError:
     |  
     |  characters_written
     |  
     |  errno
     |      POSIX exception code
     |  
     |  filename
     |      exception filename
     |  
     |  filename2
     |      second exception filename
     |  
     |  strerror
     |      exception strerror
     |  
     |  winerror
     |      Win32 exception code
     |  
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |  
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |  
     |  __repr__(self, /)
     |      Return repr(self).
     |  
     |  __setstate__(...)
     |  
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |  
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |  
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |  
     |  __cause__
     |      exception cause
     |  
     |  __context__
     |      exception context
     |  
     |  __dict__
     |  
     |  __suppress_context__
     |  
     |  __traceback__
     |  
     |  args
    
    class ConnectionResetError(ConnectionError)
     |  Connection reset.
     |  
     |  Method resolution order:
     |      ConnectionResetError
     |      ConnectionError
     |      OSError
     |      Exception
     |      BaseException
     |      object
     |  
     |  Methods defined here:
     |  
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from OSError:
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  ----------------------------------------------------------------------
     |  Static methods inherited from OSError:
     |
     |  __new__(*args, **kwargs) class method of OSError
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from OSError:
     |
     |  characters_written
     |
     |  errno
     |      POSIX exception code
     |
     |  filename
     |      exception filename
     |
     |  filename2
     |      second exception filename
     |
     |  strerror
     |      exception strerror
     |
     |  winerror
     |      Win32 exception code
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class DeprecationWarning(Warning)
     |  Base class for warnings about deprecated features.
     |
     |  Method resolution order:
     |      DeprecationWarning
     |      Warning
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
```

```
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class EOFError(Exception)
     |  Read beyond end of file.
     |
     |  Method resolution order:
     |      EOFError
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class EncodingWarning(Warning)
     |  Base class for warnings about encodings.
     |
     |  Method resolution order:
     |      EncodingWarning
```

```
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

EnvironmentError = class OSError(Exception)
 |  Base class for I/O related errors.
 |
 |  Method resolution order:
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      BlockingIOError
 |      ChildProcessError
 |      ConnectionError
 |      FileExistsError
 |      ... and 7 other subclasses
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class Exception(BaseException)
 |  Common base class for all non-exit exceptions.
 |
 |  Method resolution order:
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      ArithmeticError
 |      AssertionError
 |      AttributeError
 |      BufferError
 |      ... and 16 other subclasses
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ExceptionGroup(BaseExceptionGroup, Exception)
 |  Method resolution order:
 |      ExceptionGroup
 |      BaseExceptionGroup
 |      Exception
 |      BaseException
 |      object
 |
 |  Data descriptors defined here:
 |
 |  __weakref__
 |      list of weak references to the object
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseExceptionGroup:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  derive(...)
 |
 |  split(...)
 |
 |  subgroup(...)
 |
 |  ----------------------------------------------------------------------
 |  Class methods inherited from BaseExceptionGroup:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from BaseExceptionGroup:
 |
 |  __new__(*args, **kwargs) class method of BaseExceptionGroup
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseExceptionGroup:
 |
 |  exceptions
 |      nested exceptions
 |
 |  message
 |      exception message
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class FileExistsError(OSError)
 |  File already exists.
 |
 |  Method resolution order:
 |      FileExistsError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |  
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |  
 |  characters_written
 |  
 |  errno
 |      POSIX exception code
 |  
 |  filename
 |      exception filename
 |  
 |  filename2
 |      second exception filename
 |  
 |  strerror
 |      exception strerror
 |  
 |  winerror
 |      Win32 exception code
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args

class FileNotFoundError(OSError)
 |  File not found.
 |  
 |  Method resolution order:
 |      FileNotFoundError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |  
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |  
 |  characters_written
 |  
 |  errno
 |      POSIX exception code
 |  
 |  filename
 |      exception filename
 |  
 |  filename2
 |      second exception filename
 |  
 |  strerror
 |      exception strerror
 |  
 |  winerror
 |      Win32 exception code
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args

class FloatingPointError(ArithmeticError)
 |  Floating-point operation failed.
 |  
 |  Method resolution order:
 |      FloatingPointError
 |      ArithmeticError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class FutureWarning(Warning)
 |  Base class for warnings about constructs that will change semantically
 |  in the future.
 |
 |  Method resolution order:
 |      FutureWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class GeneratorExit(BaseException)
 |  Request that a generator exit.
 |
 |  Method resolution order:
 |      GeneratorExit
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

IOError = class OSError(Exception)
 |  Base class for I/O related errors.
 |
 |  Method resolution order:
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      BlockingIOError
 |      ChildProcessError
 |      ConnectionError
 |      FileExistsError
 |      ... and 7 other subclasses
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
```

```
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ImportError(Exception)
 |  Import can't find module, or can't find name in module.
 |
 |  Method resolution order:
 |      ImportError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      ModuleNotFoundError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  msg
 |      exception message
 |
 |  name
 |      module name
 |
 |  name_from
 |      name imported from module
 |
 |  path
 |      module path
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ImportWarning(Warning)
 |  Base class for warnings about probable mistakes in module imports
 |
 |  Method resolution order:
 |      ImportWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class IndentationError(SyntaxError)
 |  Improper indentation.
 |
 |  Method resolution order:
 |      IndentationError
 |      SyntaxError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      TabError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from SyntaxError:
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from SyntaxError:
 |
 |  end_lineno
 |      exception end lineno
 |
 |  end_offset
 |      exception end offset
 |
 |  filename
 |      exception filename
 |
 |  lineno
 |      exception lineno
 |
 |  msg
 |      exception msg
 |
 |  offset
 |      exception offset
 |
 |  print_file_and_line
 |      exception print_file_and_line
 |
 |  text
 |      exception text
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class IndexError(LookupError)
 |  Sequence index out of range.
 |
 |  Method resolution order:
 |      IndexError
 |      LookupError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
```

```
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class InterruptedError(OSError)
     |  Interrupted by signal.
     |
     |  Method resolution order:
     |      InterruptedError
     |      OSError
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from OSError:
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  ----------------------------------------------------------------------
     |  Static methods inherited from OSError:
     |
     |  __new__(*args, **kwargs) class method of OSError
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from OSError:
     |
     |  characters_written
     |
     |  errno
     |      POSIX exception code
     |
     |  filename
     |      exception filename
     |
     |  filename2
     |      second exception filename
     |
     |  strerror
     |      exception strerror
     |
     |  winerror
     |      Win32 exception code
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
```

```
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class IsADirectoryError(OSError)
 |  Operation doesn't work on directories.
 |
 |  Method resolution order:
 |      IsADirectoryError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class KeyError(LookupError)
 |  Mapping key not found.
 |
 |  Method resolution order:
 |      KeyError
 |      LookupError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from LookupError:
 |
 |  __new__(*args, **kwargs) class method of LookupError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class KeyboardInterrupt(BaseException)
 |  Program interrupted by user.
 |
 |  Method resolution order:
 |      KeyboardInterrupt
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class LookupError(Exception)
 |  Base class for lookup errors.
 |
 |  Method resolution order:
 |      LookupError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      IndexError
 |      KeyError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class MemoryError(Exception)
 |  Out of memory.
 |
 |  Method resolution order:
 |      MemoryError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ModuleNotFoundError(ImportError)
 |  Module not found.
 |
 |  Method resolution order:
 |      ModuleNotFoundError
 |      ImportError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from ImportError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from ImportError:
 |
 |  msg
 |      exception message
 |
 |  name
 |      module name
 |
 |  name_from
 |      name imported from module
 |
 |  path
 |      module path
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class NameError(Exception)
 |  Name not found globally.
 |
 |  Method resolution order:
 |      NameError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      UnboundLocalError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  name
 |      name
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class NotADirectoryError(OSError)
 |  Operation only works on directories.
 |  
 |  Method resolution order:
 |      NotADirectoryError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |  
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |  
 |  characters_written
 |  
 |  errno
 |      POSIX exception code
 |  
 |  filename
 |      exception filename
 |  
 |  filename2
 |      second exception filename
 |  
 |  strerror
 |      exception strerror
 |  
 |  winerror
 |      Win32 exception code
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args

class NotImplementedError(RuntimeError)
 |  Method or function hasn't been implemented yet.
 |  
 |  Method resolution order:
 |      NotImplementedError
 |      RuntimeError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class OSError(Exception)
 |  Base class for I/O related errors.
 |
 |  Method resolution order:
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      BlockingIOError
 |      ChildProcessError
 |      ConnectionError
 |      FileExistsError
 |      ... and 7 other subclasses
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class OverflowError(ArithmeticError)
 |  Result too large to be represented.
 |
 |  Method resolution order:
 |      OverflowError
 |      ArithmeticError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class PendingDeprecationWarning(Warning)
 |  Base class for warnings about features which will be deprecated
 |  in the future.
 |
 |  Method resolution order:
 |      PendingDeprecationWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class PermissionError(OSError)
 |  Not enough permissions.
 |
 |  Method resolution order:
 |      PermissionError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ProcessLookupError(OSError)
 |  Process not found.
 |
 |  Method resolution order:
 |      ProcessLookupError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args
    
class RecursionError(RuntimeError)
 |  Recursion limit exceeded.
 |
 |  Method resolution order:
 |      RecursionError
 |      RuntimeError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args
    
class ReferenceError(Exception)
 |  Weak ref proxy used after referent went away.
 |
 |  Method resolution order:
 |      ReferenceError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args

class ResourceWarning(Warning)
 |  Base class for warnings about resource usage.
 |  
 |  Method resolution order:
 |      ResourceWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
```

```
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class RuntimeError(Exception)
     |  Unspecified run-time error.
     |
     |  Method resolution order:
     |      RuntimeError
     |      Exception
     |      BaseException
     |      object
     |
     |  Built-in subclasses:
     |      NotImplementedError
     |      RecursionError
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class RuntimeWarning(Warning)
     |  Base class for warnings about dubious runtime behavior.
     |
     |  Method resolution order:
     |      RuntimeWarning
     |      Warning
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class StopAsyncIteration(Exception)
     |  Signal the end from iterator.__anext__().
     |
     |  Method resolution order:
     |      StopAsyncIteration
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class StopIteration(Exception)
     |  Signal the end from iterator.__next__().
     |
     |  Method resolution order:
```

```
 |      StopIteration
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  value
 |      generator return value
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class SyntaxError(Exception)
 |  Invalid syntax.
 |
 |  Method resolution order:
 |      SyntaxError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      IndentationError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  end_lineno
 |      exception end lineno
 |
 |  end_offset
 |      exception end offset
 |
 |  filename
 |      exception filename
 |
 |  lineno
 |      exception lineno
 |
 |  msg
 |      exception msg
 |
 |  offset
 |      exception offset
 |
 |  print_file_and_line
 |      exception print_file_and_line
 |
 |  text
 |      exception text
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class SyntaxWarning(Warning)
 |  Base class for warnings about dubious syntax.
 |
 |  Method resolution order:
 |      SyntaxWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args
    
class SystemError(Exception)
 |  Internal error in the Python interpreter.
 |  
 |  Please report this to the Python maintainer, along with the traceback,
 |  the Python version, and the hardware/OS platform and version.
 |  
 |  Method resolution order:
 |      SystemError
 |      Exception
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __reduce__(...)
 |      Helper for pickle.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setstate__(...)
 |  
 |  __str__(self, /)
 |      Return str(self).
 |  
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |  
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |  
 |  __cause__
 |      exception cause
 |  
 |  __context__
 |      exception context
 |  
 |  __dict__
 |  
 |  __suppress_context__
 |  
 |  __traceback__
 |  
 |  args
    
class SystemExit(BaseException)
 |  Request to exit from the interpreter.
 |  
 |  Method resolution order:
 |      SystemExit
 |      BaseException
 |      object
 |  
 |  Methods defined here:
 |  
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |  
 |  code
 |      exception code
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from BaseException:
 |
 |  __new__(*args, **kwargs) class method of BaseException
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class TabError(IndentationError)
 |  Improper mixture of spaces and tabs.
 |
 |  Method resolution order:
 |      TabError
 |      IndentationError
 |      SyntaxError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from SyntaxError:
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from SyntaxError:
 |
 |  end_lineno
 |      exception end lineno
 |
 |  end_offset
 |      exception end offset
 |
 |  filename
 |      exception filename
 |
 |  lineno
 |      exception lineno
 |
 |  msg
 |      exception msg
 |
 |  offset
 |      exception offset
 |
 |  print_file_and_line
 |      exception print_file_and_line
 |
 |  text
 |      exception text
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
```

```
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class TimeoutError(OSError)
 |  Timeout expired.
 |
 |  Method resolution order:
 |      TimeoutError
 |      OSError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from OSError:
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from OSError:
 |
 |  __new__(*args, **kwargs) class method of OSError
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from OSError:
 |
 |  characters_written
 |
 |  errno
 |      POSIX exception code
 |
 |  filename
 |      exception filename
 |
 |  filename2
 |      second exception filename
 |
 |  strerror
 |      exception strerror
 |
 |  winerror
 |      Win32 exception code
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class TypeError(Exception)
 |  Inappropriate argument type.
 |
 |  Method resolution order:
 |      TypeError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class UnboundLocalError(NameError)
 |  Local name referenced but not bound to a value.
 |
 |  Method resolution order:
 |      UnboundLocalError
 |      NameError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from NameError:
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from NameError:
 |
 |  name
 |      name
 |
 |  ----------------------------------------------------------------------
 |  Static methods inherited from Exception:
 |
 |  __new__(*args, **kwargs) class method of Exception
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class UnicodeDecodeError(UnicodeError)
 |  Unicode decoding error.
 |
 |  Method resolution order:
 |      UnicodeDecodeError
 |      UnicodeError
 |      ValueError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  encoding
 |      exception encoding
 |
 |  end
 |      exception end
 |
 |  object
 |      exception object
 |
 |  reason
 |      exception reason
 |
 |  start
 |      exception start
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class UnicodeEncodeError(UnicodeError)
 |  Unicode encoding error.
 |
 |  Method resolution order:
 |      UnicodeEncodeError
 |      UnicodeError
 |      ValueError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  encoding
 |      exception encoding
 |
 |  end
 |      exception end
 |
 |  object
 |      exception object
 |
 |  reason
 |      exception reason
 |
 |  start
 |      exception start
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
```

```
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class UnicodeError(ValueError)
     |  Unicode related error.
     |
     |  Method resolution order:
     |      UnicodeError
     |      ValueError
     |      Exception
     |      BaseException
     |      object
     |
     |  Built-in subclasses:
     |      UnicodeDecodeError
     |      UnicodeEncodeError
     |      UnicodeTranslateError
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    class UnicodeTranslateError(UnicodeError)
     |  Unicode translation error.
     |
     |  Method resolution order:
     |      UnicodeTranslateError
     |      UnicodeError
     |      ValueError
     |      Exception
     |      BaseException
     |      object
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
```

```
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  encoding
 |      exception encoding
 |
 |  end
 |      exception end
 |
 |  object
 |      exception object
 |
 |  reason
 |      exception reason
 |
 |  start
 |      exception start
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class UnicodeWarning(Warning)
 |  Base class for warnings about Unicode related problems, mostly
 |  related to conversion problems.
 |
 |  Method resolution order:
 |      UnicodeWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class UserWarning(Warning)
 |  Base class for warnings generated by user code.
 |
 |  Method resolution order:
 |      UserWarning
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class ValueError(Exception)
 |  Inappropriate argument value (of correct type).
 |
 |  Method resolution order:
 |      ValueError
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      UnicodeError
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class Warning(Exception)
 |  Base class for warning categories.
 |
 |  Method resolution order:
 |      Warning
 |      Exception
 |      BaseException
 |      object
 |
 |  Built-in subclasses:
 |      BytesWarning
 |      DeprecationWarning
 |      EncodingWarning
 |      FutureWarning
 |      ... and 7 other subclasses
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
```

```
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    WindowsError = class OSError(Exception)
     |  Base class for I/O related errors.
     |
     |  Method resolution order:
     |      OSError
     |      Exception
     |      BaseException
     |      object
     |
     |  Built-in subclasses:
     |      BlockingIOError
     |      ChildProcessError
     |      ConnectionError
     |      FileExistsError
     |      ... and 7 other subclasses
     |
     |  Methods defined here:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  __reduce__(...)
     |      Helper for pickle.
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors defined here:
     |
     |  characters_written
     |
     |  errno
     |      POSIX exception code
     |
     |  filename
     |      exception filename
     |
     |  filename2
     |      second exception filename
     |
     |  strerror
     |      exception strerror
     |
     |  winerror
     |      Win32 exception code
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from BaseException:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setstate__(...)
     |
     |  add_note(...)
     |      Exception.add_note(note) --
     |      add a note to the exception
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
```

```
 |
 |  args

class ZeroDivisionError(ArithmeticError)
 |  Second argument to a division or modulo operation was zero.
 |
 |  Method resolution order:
 |      ZeroDivisionError
 |      ArithmeticError
 |      Exception
 |      BaseException
 |      object
 |
 |  Methods defined here:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from BaseException:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  add_note(...)
 |      Exception.add_note(note) --
 |      add a note to the exception
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class bool(int)
 |  bool(x) -> bool
 |
 |  Returns True when the argument x is true, False otherwise.
 |  The builtins True and False are the only two instances of the class bool.
 |  The class bool is a subclass of the class int, and cannot be subclassed.
 |
 |  Method resolution order:
 |      bool
 |      int
 |      object
 |
 |  Methods defined here:
 |
 |  __and__(self, value, /)
 |      Return self&value.
 |
 |  __invert__(self, /)
 |      ~self
 |
 |  __or__(self, value, /)
 |      Return self|value.
 |
 |  __rand__(self, value, /)
 |      Return value&self.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __ror__(self, value, /)
 |      Return value|self.
 |
 |  __rxor__(self, value, /)
 |      Return value^self.
 |
 |  __xor__(self, value, /)
 |      Return self^value.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Methods inherited from int:
 |  
 |  __abs__(self, /)
 |      abs(self)
 |  
 |  __add__(self, value, /)
 |      Return self+value.
 |  
 |  __bool__(self, /)
 |      True if self else False
 |  
 |  __ceil__(...)
 |      Ceiling of an Integral returns itself.
 |  
 |  __divmod__(self, value, /)
 |      Return divmod(self, value).
 |  
 |  __eq__(self, value, /)
 |      Return self==value.
 |  
 |  __float__(self, /)
 |      float(self)
 |  
 |  __floor__(...)
 |      Flooring an Integral returns itself.
 |  
 |  __floordiv__(self, value, /)
 |      Return self//value.
 |  
 |  __format__(self, format_spec, /)
 |      Convert to a string 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).
 |  
 |  __index__(self, /)
 |      Return self converted to an integer, if self is suitable for use as an index into a list.
 |  
 |  __int__(self, /)
 |      int(self)
 |  
 |  __le__(self, value, /)
 |      Return self<=value.
 |  
 |  __lshift__(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).
 |  
 |  __rfloordiv__(self, value, /)
 |      Return value//self.
 |  
 |  __rlshift__(self, value, /)
 |      Return value<<self.
 |  
 |  __rmod__(self, value, /)
 |      Return value%self.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __round__(...)
 |      Rounding an Integral returns itself.
 |
 |      Rounding with an ndigits argument also returns an integer.
 |
 |  __rpow__(self, value, mod=None, /)
 |      Return pow(value, self, mod).
 |
 |  __rrshift__(self, value, /)
 |      Return value>>self.
 |
 |  __rshift__(self, value, /)
 |      Return self>>value.
 |
 |  __rsub__(self, value, /)
 |      Return value-self.
 |
 |  __rtruediv__(self, value, /)
 |      Return value/self.
 |
 |  __sizeof__(self, /)
 |      Returns size in memory, in bytes.
 |
 |  __sub__(self, value, /)
 |      Return self-value.
 |
 |  __truediv__(self, value, /)
 |      Return self/value.
 |
 |  __trunc__(...)
 |      Truncating an Integral returns itself.
 |
 |  as_integer_ratio(self, /)
 |      Return a pair of integers, whose ratio is equal to the original int.
 |
 |      The ratio is in lowest terms and has a positive denominator.
 |
 |      >>> (10).as_integer_ratio()
 |      (10, 1)
 |      >>> (-10).as_integer_ratio()
 |      (-10, 1)
 |      >>> (0).as_integer_ratio()
 |      (0, 1)
 |
 |  bit_count(self, /)
 |      Number of ones in the binary representation of the absolute value of self.
 |
 |      Also known as the population count.
 |
 |      >>> bin(13)
 |      '0b1101'
 |      >>> (13).bit_count()
 |      3
 |
 |  bit_length(self, /)
 |      Number of bits necessary to represent self in binary.
 |
 |      >>> bin(37)
 |      '0b100101'
 |      >>> (37).bit_length()
 |      6
 |
 |  conjugate(...)
 |      Returns self, the complex conjugate of any int.
 |
 |  is_integer(self, /)
 |      Returns True. Exists for duck type compatibility with float.is_integer.
 |
 |  to_bytes(self, /, length=1, byteorder='big', *, signed=False)
 |      Return an array of bytes representing an integer.
 |
 |      length
 |        Length of bytes object to use.  An OverflowError is raised if the
 |        integer is not representable with the given number of bytes.  Default
 |        is length 1.
 |      byteorder
 |        The byte order used to represent the integer.  If byteorder is 'big',
 |        the most significant byte is at the beginning of the byte array.  If
 |        byteorder is 'little', the most significant byte is at the end of the
 |        byte array.  To request the native byte order of the host system, use
 |        `sys.byteorder' as the byte order value.  Default is to use 'big'.
 |      signed
 |        Determines whether two's complement is used to represent the integer.
 |        If signed is False and a negative integer is given, an OverflowError
 |        is raised.
 |
 |  ----------------------------------------------------------------------
 |  Class methods inherited from int:
 |
 |  from_bytes(bytes, byteorder='big', *, signed=False)
 |      Return the integer represented by the given array of bytes.
 |
 |      bytes
 |        Holds the array of bytes to convert.  The argument must either
 |        support the buffer protocol or be an iterable object producing bytes.
 |        Bytes and bytearray are examples of built-in objects that support the
 |        buffer protocol.
 |      byteorder
 |        The byte order used to represent the integer.  If byteorder is 'big',
 |        the most significant byte is at the beginning of the byte array.  If
 |        byteorder is 'little', the most significant byte is at the end of the
 |        byte array.  To request the native byte order of the host system, use
 |        `sys.byteorder' as the byte order value.  Default is to use 'big'.
 |      signed
 |        Indicates whether two's complement is used to represent the integer.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from int:
 |
 |  denominator
 |      the denominator of a rational number in lowest terms
 |
 |  imag
 |      the imaginary part of a complex number
 |
 |  numerator
 |      the numerator of a rational number in lowest terms
 |
 |  real
 |      the real part of a complex number

class bytearray(object)
 |  bytearray(iterable_of_ints) -> bytearray
 |  bytearray(string, encoding[, errors]) -> bytearray
 |  bytearray(bytes_or_buffer) -> mutable copy of bytes_or_buffer
 |  bytearray(int) -> bytes array of size given by the parameter initialized with null bytes
 |  bytearray() -> empty bytes array
 |
 |  Construct a mutable bytearray object from:
 |    - an iterable yielding integers in range(256)
 |    - a text string encoded using the specified encoding
 |    - a bytes or a buffer object
 |    - any object implementing the buffer API.
 |    - an integer
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __alloc__(...)
 |      B.__alloc__() -> int
 |
 |      Return the number of bytes actually allocated.
 |
 |  __buffer__(self, flags, /)
 |      Return a buffer object that exposes the underlying memory of the object.
 |
 |  __contains__(self, key, /)
 |      Return bool(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__(self, key, /)
 |      Return self[key].
 |
 |  __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.
 |
 |  __mod__(self, value, /)
 |      Return self%value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __reduce__(self, /)
 |      Return state information for pickling.
 |
 |  __reduce_ex__(self, proto=0, /)
 |      Return state information for pickling.
 |
 |  __release_buffer__(self, buffer, /)
 |      Release the buffer object that exposes the underlying memory of the object.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rmod__(self, value, /)
 |      Return value%self.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(self, /)
 |      Returns the size of the bytearray object in memory, in bytes.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  append(self, item, /)
 |      Append a single item to the end of the bytearray.
 |
 |      item
 |        The item to be appended.
 |
 |  capitalize(...)
 |      B.capitalize() -> copy of B
 |
 |      Return a copy of B with only its first character capitalized (ASCII)
 |      and the rest lower-cased.
 |
 |  center(self, width, fillchar=b' ', /)
 |      Return a centered string of length width.
 |
 |      Padding is done using the specified fill character.
 |
 |  clear(self, /)
 |      Remove all items from the bytearray.
 |
 |  copy(self, /)
 |      Return a copy of B.
 |
 |  count(...)
 |      B.count(sub[, start[, end]]) -> int
 |
 |      Return the number of non-overlapping occurrences of subsection sub in
 |      bytes B[start:end].  Optional arguments start and end are interpreted
 |      as in slice notation.
 |
 |  decode(self, /, encoding='utf-8', errors='strict')
 |      Decode the bytearray using the codec registered for encoding.
 |
 |      encoding
 |        The encoding with which to decode the bytearray.
 |      errors
 |        The error handling scheme to use for the handling of decoding errors.
 |        The default is 'strict' meaning that decoding errors raise a
 |        UnicodeDecodeError. Other possible values are 'ignore' and 'replace'
 |        as well as any other name registered with codecs.register_error that
 |        can handle UnicodeDecodeErrors.
 |
 |  endswith(...)
 |      B.endswith(suffix[, start[, end]]) -> bool
 |
 |      Return True if B ends with the specified suffix, False otherwise.
 |      With optional start, test B beginning at that position.
 |      With optional end, stop comparing B at that position.
 |      suffix can also be a tuple of bytes to try.
 |
 |  expandtabs(self, /, tabsize=8)
 |      Return a copy where all tab characters are expanded using spaces.
 |
 |      If tabsize is not given, a tab size of 8 characters is assumed.
 |
 |  extend(self, iterable_of_ints, /)
 |      Append all the items from the iterator or sequence to the end of the bytearray.
 |
 |      iterable_of_ints
 |        The iterable of items to append.
 |
 |  find(...)
 |      B.find(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Return -1 on failure.
 |
 |  hex(...)
 |      Create a string of hexadecimal numbers from a bytearray object.
 |
 |        sep
 |          An optional single character or byte to separate hex bytes.
 |        bytes_per_sep
 |          How many bytes between separators.  Positive values count from the
 |          right, negative values count from the left.
 |
 |      Example:
 |      >>> value = bytearray([0xb9, 0x01, 0xef])
 |      >>> value.hex()
 |      'b901ef'
 |      >>> value.hex(':')
 |      'b9:01:ef'
 |      >>> value.hex(':', 2)
 |      'b9:01ef'
 |      >>> value.hex(':', -2)
 |      'b901:ef'
 |
 |  index(...)
 |      B.index(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Raises ValueError when the subsection is not found.
 |
 |  insert(self, index, item, /)
 |      Insert a single item into the bytearray before the given index.
 |
 |      index
 |        The index where the value is to be inserted.
 |      item
 |        The item to be inserted.
 |
 |  isalnum(...)
 |      B.isalnum() -> bool
 |
 |      Return True if all characters in B are alphanumeric
 |      and there is at least one character in B, False otherwise.
 |
 |  isalpha(...)
 |      B.isalpha() -> bool
 |
 |      Return True if all characters in B are alphabetic
 |      and there is at least one character in B, False otherwise.
 |
 |  isascii(...)
 |      B.isascii() -> bool
 |
 |      Return True if B is empty or all characters in B are ASCII,
 |      False otherwise.
 |
 |  isdigit(...)
 |      B.isdigit() -> bool
 |
 |      Return True if all characters in B are digits
 |      and there is at least one character in B, False otherwise.
 |
 |  islower(...)
 |      B.islower() -> bool
 |
 |      Return True if all cased characters in B are lowercase and there is
 |      at least one cased character in B, False otherwise.
 |
 |  isspace(...)
 |      B.isspace() -> bool
 |
 |      Return True if all characters in B are whitespace
 |      and there is at least one character in B, False otherwise.
 |
 |  istitle(...)
 |      B.istitle() -> bool
 |
 |      Return True if B is a titlecased string and there is at least one
 |      character in B, i.e. uppercase characters may only follow uncased
 |      characters and lowercase characters only cased ones. Return False
 |      otherwise.
 |
 |  isupper(...)
 |      B.isupper() -> bool
 |
 |      Return True if all cased characters in B are uppercase and there is
 |      at least one cased character in B, False otherwise.
 |
 |  join(self, iterable_of_bytes, /)
 |      Concatenate any number of bytes/bytearray objects.
 |
 |      The bytearray whose method is called is inserted in between each pair.
 |
 |      The result is returned as a new bytearray object.
 |
 |  ljust(self, width, fillchar=b' ', /)
 |      Return a left-justified string of length width.
 |
 |      Padding is done using the specified fill character.
 |
 |  lower(...)
 |      B.lower() -> copy of B
 |
 |      Return a copy of B with all ASCII characters converted to lowercase.
 |
 |  lstrip(self, bytes=None, /)
 |      Strip leading bytes contained in the argument.
 |
 |      If the argument is omitted or None, strip leading ASCII whitespace.
 |
 |  partition(self, sep, /)
 |      Partition the bytearray into three parts using the given separator.
 |
 |      This will search for the separator sep in the bytearray. If the separator is
 |      found, returns a 3-tuple containing the part before the separator, the
 |      separator itself, and the part after it as new bytearray objects.
 |
 |      If the separator is not found, returns a 3-tuple containing the copy of the
 |      original bytearray object and two empty bytearray objects.
 |
 |  pop(self, index=-1, /)
 |      Remove and return a single item from B.
 |
 |        index
 |          The index from where to remove the item.
 |          -1 (the default value) means remove the last item.
 |
 |      If no index argument is given, will pop the last item.
 |
 |  remove(self, value, /)
 |      Remove the first occurrence of a value in the bytearray.
 |
 |      value
 |        The value to remove.
 |
 |  removeprefix(self, prefix, /)
 |      Return a bytearray with the given prefix string removed if present.
 |
 |      If the bytearray starts with the prefix string, return
 |      bytearray[len(prefix):].  Otherwise, return a copy of the original
 |      bytearray.
 |
 |  removesuffix(self, suffix, /)
 |      Return a bytearray with the given suffix string removed if present.
 |
 |      If the bytearray ends with the suffix string and that suffix is not
 |      empty, return bytearray[:-len(suffix)].  Otherwise, return a copy of
 |      the original bytearray.
 |
 |  replace(self, old, new, count=-1, /)
 |      Return a copy with all occurrences of substring old replaced by new.
 |
 |        count
 |          Maximum number of occurrences to replace.
 |          -1 (the default value) means replace all occurrences.
 |
 |      If the optional argument count is given, only the first count occurrences are
 |      replaced.
 |
 |  reverse(self, /)
 |      Reverse the order of the values in B in place.
 |
 |  rfind(...)
 |      B.rfind(sub[, start[, end]]) -> int
 |
 |      Return the highest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Return -1 on failure.
 |
 |  rindex(...)
 |      B.rindex(sub[, start[, end]]) -> int
 |
 |      Return the highest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Raise ValueError when the subsection is not found.
 |
 |  rjust(self, width, fillchar=b' ', /)
 |      Return a right-justified string of length width.
 |
 |      Padding is done using the specified fill character.
 |
 |  rpartition(self, sep, /)
 |      Partition the bytearray into three parts using the given separator.
 |
 |      This will search for the separator sep in the bytearray, starting at the end.
 |      If the separator is found, returns a 3-tuple containing the part before the
 |      separator, the separator itself, and the part after it as new bytearray
 |      objects.
 |
 |      If the separator is not found, returns a 3-tuple containing two empty bytearray
 |      objects and the copy of the original bytearray object.
 |
 |  rsplit(self, /, sep=None, maxsplit=-1)
 |      Return a list of the sections in the bytearray, using sep as the delimiter.
 |
 |        sep
 |          The delimiter according which to split the bytearray.
 |          None (the default value) means split on ASCII whitespace characters
 |          (space, tab, return, newline, formfeed, vertical tab).
 |        maxsplit
 |          Maximum number of splits to do.
 |          -1 (the default value) means no limit.
 |
 |      Splitting is done starting at the end of the bytearray and working to the front.
 |
 |  rstrip(self, bytes=None, /)
 |      Strip trailing bytes contained in the argument.
 |
 |      If the argument is omitted or None, strip trailing ASCII whitespace.
 |
 |  split(self, /, sep=None, maxsplit=-1)
 |      Return a list of the sections in the bytearray, using sep as the delimiter.
 |
 |      sep
 |        The delimiter according which to split the bytearray.
 |        None (the default value) means split on ASCII whitespace characters
 |        (space, tab, return, newline, formfeed, vertical tab).
 |      maxsplit
 |        Maximum number of splits to do.
 |        -1 (the default value) means no limit.
 |
 |  splitlines(self, /, keepends=False)
 |      Return a list of the lines in the bytearray, breaking at line boundaries.
 |
 |      Line breaks are not included in the resulting list unless keepends is given and
 |      true.
 |
 |  startswith(...)
 |      B.startswith(prefix[, start[, end]]) -> bool
 |
 |      Return True if B starts with the specified prefix, False otherwise.
 |      With optional start, test B beginning at that position.
 |      With optional end, stop comparing B at that position.
 |      prefix can also be a tuple of bytes to try.
 |
 |  strip(self, bytes=None, /)
 |      Strip leading and trailing bytes contained in the argument.
 |
 |      If the argument is omitted or None, strip leading and trailing ASCII whitespace.
 |
 |  swapcase(...)
 |      B.swapcase() -> copy of B
 |
 |      Return a copy of B with uppercase ASCII characters converted
 |      to lowercase ASCII and vice versa.
 |
 |  title(...)
 |      B.title() -> copy of B
 |
 |      Return a titlecased version of B, i.e. ASCII words start with uppercase
 |      characters, all remaining cased characters have lowercase.
 |
 |  translate(self, table, /, delete=b'')
 |      Return a copy with each character mapped by the given translation table.
 |
 |        table
 |          Translation table, which must be a bytes object of length 256.
 |
 |      All characters occurring in the optional argument delete are removed.
 |      The remaining characters are mapped through the given translation table.
 |
 |  upper(...)
 |      B.upper() -> copy of B
 |
 |      Return a copy of B with all ASCII characters converted to uppercase.
 |
 |  zfill(self, width, /)
 |      Pad a numeric string with zeros on the left, to fill a field of the given width.
 |
 |      The original string is never truncated.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  fromhex(string, /)
 |      Create a bytearray object from a string of hexadecimal numbers.
 |
 |      Spaces between two numbers are accepted.
 |      Example: bytearray.fromhex('B9 01EF') -> bytearray(b'\\xb9\\x01\\xef')
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  maketrans(frm, to, /)
 |      Return a translation table useable for the bytes or bytearray translate method.
 |
 |      The returned table will be one where each byte in frm is mapped to the byte at
 |      the same position in to.
 |
 |      The bytes objects frm and to must be of the same length.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None

class bytes(object)
 |  bytes(iterable_of_ints) -> bytes
 |  bytes(string, encoding[, errors]) -> bytes
 |  bytes(bytes_or_buffer) -> immutable copy of bytes_or_buffer
 |  bytes(int) -> bytes object of size given by the parameter initialized with null bytes
 |  bytes() -> empty bytes object
 |
 |  Construct an immutable array of bytes from:
 |    - an iterable yielding integers in range(256)
 |    - a text string encoded using the specified encoding
 |    - any object implementing the buffer API.
 |    - an integer
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __buffer__(self, flags, /)
 |      Return a buffer object that exposes the underlying memory of the object.
 |
 |  __bytes__(self, /)
 |      Convert this value to exact type bytes.
 |
 |  __contains__(self, key, /)
 |      Return bool(key in self).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
 |
 |  __getnewargs__(...)
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __mod__(self, value, /)
 |      Return self%value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rmod__(self, value, /)
 |      Return value%self.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  capitalize(...)
 |      B.capitalize() -> copy of B
 |
 |      Return a copy of B with only its first character capitalized (ASCII)
 |      and the rest lower-cased.
 |
 |  center(self, width, fillchar=b' ', /)
 |      Return a centered string of length width.
 |
 |      Padding is done using the specified fill character.
 |
 |  count(...)
 |      B.count(sub[, start[, end]]) -> int
 |
 |      Return the number of non-overlapping occurrences of subsection sub in
 |      bytes B[start:end].  Optional arguments start and end are interpreted
 |      as in slice notation.
 |
 |  decode(self, /, encoding='utf-8', errors='strict')
 |      Decode the bytes using the codec registered for encoding.
 |
 |      encoding
 |        The encoding with which to decode the bytes.
 |      errors
 |        The error handling scheme to use for the handling of decoding errors.
 |        The default is 'strict' meaning that decoding errors raise a
 |        UnicodeDecodeError. Other possible values are 'ignore' and 'replace'
 |        as well as any other name registered with codecs.register_error that
 |        can handle UnicodeDecodeErrors.
 |
 |  endswith(...)
 |      B.endswith(suffix[, start[, end]]) -> bool
 |
 |      Return True if B ends with the specified suffix, False otherwise.
 |      With optional start, test B beginning at that position.
 |      With optional end, stop comparing B at that position.
 |      suffix can also be a tuple of bytes to try.
 |
 |  expandtabs(self, /, tabsize=8)
 |      Return a copy where all tab characters are expanded using spaces.
 |
 |      If tabsize is not given, a tab size of 8 characters is assumed.
 |
 |  find(...)
 |      B.find(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Return -1 on failure.
 |
 |  hex(...)
 |      Create a string of hexadecimal numbers from a bytes object.
 |
 |        sep
 |          An optional single character or byte to separate hex bytes.
 |        bytes_per_sep
 |          How many bytes between separators.  Positive values count from the
 |          right, negative values count from the left.
 |
 |      Example:
 |      >>> value = b'\xb9\x01\xef'
 |      >>> value.hex()
 |      'b901ef'
 |      >>> value.hex(':')
 |      'b9:01:ef'
 |      >>> value.hex(':', 2)
 |      'b9:01ef'
 |      >>> value.hex(':', -2)
 |      'b901:ef'
 |
 |  index(...)
 |      B.index(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Raises ValueError when the subsection is not found.
 |
 |  isalnum(...)
 |      B.isalnum() -> bool
 |
 |      Return True if all characters in B are alphanumeric
 |      and there is at least one character in B, False otherwise.
 |
 |  isalpha(...)
 |      B.isalpha() -> bool
 |
 |      Return True if all characters in B are alphabetic
 |      and there is at least one character in B, False otherwise.
 |
 |  isascii(...)
 |      B.isascii() -> bool
 |
 |      Return True if B is empty or all characters in B are ASCII,
 |      False otherwise.
 |
 |  isdigit(...)
 |      B.isdigit() -> bool
 |
 |      Return True if all characters in B are digits
 |      and there is at least one character in B, False otherwise.
 |
 |  islower(...)
 |      B.islower() -> bool
 |
 |      Return True if all cased characters in B are lowercase and there is
 |      at least one cased character in B, False otherwise.
 |
 |  isspace(...)
 |      B.isspace() -> bool
 |
 |      Return True if all characters in B are whitespace
 |      and there is at least one character in B, False otherwise.
 |
 |  istitle(...)
 |      B.istitle() -> bool
 |
 |      Return True if B is a titlecased string and there is at least one
 |      character in B, i.e. uppercase characters may only follow uncased
 |      characters and lowercase characters only cased ones. Return False
 |      otherwise.
 |
 |  isupper(...)
 |      B.isupper() -> bool
 |
 |      Return True if all cased characters in B are uppercase and there is
 |      at least one cased character in B, False otherwise.
 |
 |  join(self, iterable_of_bytes, /)
 |      Concatenate any number of bytes objects.
 |
 |      The bytes whose method is called is inserted in between each pair.
 |      
 |      The result is returned as a new bytes object.
 |      
 |      Example: b'.'.join([b'ab', b'pq', b'rs']) -> b'ab.pq.rs'.
 |  
 |  ljust(self, width, fillchar=b' ', /)
 |      Return a left-justified string of length width.
 |      
 |      Padding is done using the specified fill character.
 |  
 |  lower(...)
 |      B.lower() -> copy of B
 |      
 |      Return a copy of B with all ASCII characters converted to lowercase.
 |  
 |  lstrip(self, bytes=None, /)
 |      Strip leading bytes contained in the argument.
 |      
 |      If the argument is omitted or None, strip leading  ASCII whitespace.
 |  
 |  partition(self, sep, /)
 |      Partition the bytes into three parts using the given separator.
 |      
 |      This will search for the separator sep in the bytes. If the separator is found,
 |      returns a 3-tuple containing the part before the separator, the separator
 |      itself, and the part after it.
 |      
 |      If the separator is not found, returns a 3-tuple containing the original bytes
 |      object and two empty bytes objects.
 |  
 |  removeprefix(self, prefix, /)
 |      Return a bytes object with the given prefix string removed if present.
 |      
 |      If the bytes starts with the prefix string, return bytes[len(prefix):].
 |      Otherwise, return a copy of the original bytes.
 |  
 |  removesuffix(self, suffix, /)
 |      Return a bytes object with the given suffix string removed if present.
 |      
 |      If the bytes ends with the suffix string and that suffix is not empty,
 |      return bytes[:-len(prefix)].  Otherwise, return a copy of the original
 |      bytes.
 |  
 |  replace(self, old, new, count=-1, /)
 |      Return a copy with all occurrences of substring old replaced by new.
 |      
 |        count
 |          Maximum number of occurrences to replace.
 |          -1 (the default value) means replace all occurrences.
 |      
 |      If the optional argument count is given, only the first count occurrences are
 |      replaced.
 |  
 |  rfind(...)
 |      B.rfind(sub[, start[, end]]) -> int
 |      
 |      Return the highest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |      
 |      Return -1 on failure.
 |  
 |  rindex(...)
 |      B.rindex(sub[, start[, end]]) -> int
 |      
 |      Return the highest index in B where subsection sub is found,
 |      such that sub is contained within B[start,end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |      
 |      Raise ValueError when the subsection is not found.
 |  
 |  rjust(self, width, fillchar=b' ', /)
 |      Return a right-justified string of length width.
 |      
 |      Padding is done using the specified fill character.
 |  
 |  rpartition(self, sep, /)
 |      Partition the bytes into three parts using the given separator.
 |      
 |      This will search for the separator sep in the bytes, starting at the end. If
 |      the separator is found, returns a 3-tuple containing the part before the
 |      separator, the separator itself, and the part after it.
 |      
 |      If the separator is not found, returns a 3-tuple containing two empty bytes
 |      objects and the original bytes object.
 |  
 |  rsplit(self, /, sep=None, maxsplit=-1)
 |      Return a list of the sections in the bytes, using sep as the delimiter.
 |      
 |        sep
 |          The delimiter according which to split the bytes.
 |          None (the default value) means split on ASCII whitespace characters
 |          (space, tab, return, newline, formfeed, vertical tab).
 |        maxsplit
 |          Maximum number of splits to do.
 |          -1 (the default value) means no limit.
 |      
 |      Splitting is done starting at the end of the bytes and working to the front.
 |  
 |  rstrip(self, bytes=None, /)
 |      Strip trailing bytes contained in the argument.
 |
 |      If the argument is omitted or None, strip trailing ASCII whitespace.
 |
 |  split(self, /, sep=None, maxsplit=-1)
 |      Return a list of the sections in the bytes, using sep as the delimiter.
 |
 |      sep
 |        The delimiter according which to split the bytes.
 |        None (the default value) means split on ASCII whitespace characters
 |        (space, tab, return, newline, formfeed, vertical tab).
 |      maxsplit
 |        Maximum number of splits to do.
 |        -1 (the default value) means no limit.
 |
 |  splitlines(self, /, keepends=False)
 |      Return a list of the lines in the bytes, breaking at line boundaries.
 |
 |      Line breaks are not included in the resulting list unless keepends is given and
 |      true.
 |
 |  startswith(...)
 |      B.startswith(prefix[, start[, end]]) -> bool
 |
 |      Return True if B starts with the specified prefix, False otherwise.
 |      With optional start, test B beginning at that position.
 |      With optional end, stop comparing B at that position.
 |      prefix can also be a tuple of bytes to try.
 |
 |  strip(self, bytes=None, /)
 |      Strip leading and trailing bytes contained in the argument.
 |
 |      If the argument is omitted or None, strip leading and trailing ASCII whitespace.
 |
 |  swapcase(...)
 |      B.swapcase() -> copy of B
 |
 |      Return a copy of B with uppercase ASCII characters converted
 |      to lowercase ASCII and vice versa.
 |
 |  title(...)
 |      B.title() -> copy of B
 |
 |      Return a titlecased version of B, i.e. ASCII words start with uppercase
 |      characters, all remaining cased characters have lowercase.
 |
 |  translate(self, table, /, delete=b'')
 |      Return a copy with each character mapped by the given translation table.
 |
 |        table
 |          Translation table, which must be a bytes object of length 256.
 |
 |      All characters occurring in the optional argument delete are removed.
 |      The remaining characters are mapped through the given translation table.
 |
 |  upper(...)
 |      B.upper() -> copy of B
 |
 |      Return a copy of B with all ASCII characters converted to uppercase.
 |
 |  zfill(self, width, /)
 |      Pad a numeric string with zeros on the left, to fill a field of the given width.
 |
 |      The original string is never truncated.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  fromhex(string, /)
 |      Create a bytes object from a string of hexadecimal numbers.
 |
 |      Spaces between two numbers are accepted.
 |      Example: bytes.fromhex('B9 01EF') -> b'\\xb9\\x01\\xef'.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  maketrans(frm, to, /)
 |      Return a translation table useable for the bytes or bytearray translate method.
 |
 |      The returned table will be one where each byte in frm is mapped to the byte at
 |      the same position in to.
 |
 |      The bytes objects frm and to must be of the same length.

class classmethod(object)
 |  classmethod(function) -> method
 |
 |  Convert a function to be a class method.
 |
 |  A class method receives the class as implicit first argument,
 |  just like an instance method receives the instance.
 |  To declare a class method, use this idiom:
 |
 |    class C:
 |        @classmethod
 |        def f(cls, arg1, arg2, argN):
 |            ...
 |
 |  It can be called either on the class (e.g. C.f()) or on an instance
 |  (e.g. C().f()).  The instance is ignored except for its class.
 |  If a class method is called for a derived class, the derived class
 |  object is passed as the implied first argument.
 |
 |  Class methods are different than C++ or Java static methods.
 |  If you want those, see the staticmethod builtin.
 |
 |  Methods defined here:
 |
 |  __get__(self, instance, owner=None, /)
 |      Return an attribute of instance, which is of type owner.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |
 |  __func__
 |
 |  __isabstractmethod__
 |
 |  __wrapped__

class complex(object)
 |  complex(real=0, imag=0)
 |
 |  Create a complex number from a string or numbers.
 |
 |  If a string is given, parse it as a complex number.
 |  If a single number is given, convert it to a complex number.
 |  If the 'real' or 'imag' arguments are given, create a complex number
 |  with the specified real and imaginary components.
 |
 |  Methods defined here:
 |
 |  __abs__(self, /)
 |      abs(self)
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __bool__(self, /)
 |      True if self else False
 |
 |  __complex__(self, /)
 |      Convert this value to exact type complex.
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __format__(self, format_spec, /)
 |      Convert to a string 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).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __lt__(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.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __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.
 |
 |  conjugate(self, /)
 |      Return the complex conjugate of its argument. (3-4j).conjugate() == 3+4j.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      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

class dict(object)
 |  dict() -> new empty dictionary
 |  dict(mapping) -> new dictionary initialized from a mapping object's
 |      (key, value) pairs
 |  dict(iterable) -> new dictionary initialized as if via:
 |      d = {}
 |      for k, v in iterable:
 |          d[k] = v
 |  dict(**kwargs) -> new dictionary initialized with the name=value pairs
 |      in the keyword argument list.  For example:  dict(one=1, two=2)
 |
 |  Built-in subclasses:
 |      StgDict
 |
 |  Methods defined here:
 |
 |  __contains__(self, key, /)
 |      True if the dictionary has the specified key, else False.
 |
 |  __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__(self, key, /)
 |      Return self[key].
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __ior__(self, value, /)
 |      Return self|=value.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __or__(self, value, /)
 |      Return self|value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __reversed__(self, /)
 |      Return a reverse iterator over the dict keys.
 |
 |  __ror__(self, value, /)
 |      Return value|self.
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(...)
 |      D.__sizeof__() -> size of D in memory, in bytes
 |
 |  clear(...)
 |      D.clear() -> None.  Remove all items from D.
 |
 |  copy(...)
 |      D.copy() -> a shallow copy of D
 |
 |  get(self, key, default=None, /)
 |      Return the value for key if key is in the dictionary, else default.
 |
 |  items(...)
 |      D.items() -> a set-like object providing a view on D's items
 |
 |  keys(...)
 |      D.keys() -> a set-like object providing a view on D's keys
 |
 |  pop(...)
 |      D.pop(k[,d]) -> v, remove specified key and return the corresponding value.
 |
 |      If the key is not found, return the default if given; otherwise,
 |      raise a KeyError.
 |
 |  popitem(self, /)
 |      Remove and return a (key, value) pair as a 2-tuple.
 |
 |      Pairs are returned in LIFO (last-in, first-out) order.
 |      Raises KeyError if the dict is empty.
 |
 |  setdefault(self, key, default=None, /)
 |      Insert key with a value of default if key is not in the dictionary.
 |
 |      Return the value for key if key is in the dictionary, else default.
 |
 |  update(...)
 |      D.update([E, ]**F) -> None.  Update D from dict/iterable E and F.
 |      If E is present and has a .keys() method, then does:  for k in E: D[k] = E[k]
 |      If E is present and lacks a .keys() method, then does:  for k, v in E: D[k] = v
 |      In either case, this is followed by: for k in F:  D[k] = F[k]
 |
 |  values(...)
 |      D.values() -> an object providing a view on D's values
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  fromkeys(iterable, value=None, /)
 |      Create a new dictionary with keys from iterable and values set to value.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None

class enumerate(object)
 |  enumerate(iterable, start=0)
 |
 |  Return an enumerate object.
 |
 |    iterable
 |      an object supporting iteration
 |
 |  The enumerate object yields pairs containing a count (from start, which
 |  defaults to zero) and a value yielded by the iterable argument.
 |
 |  enumerate is useful for obtaining an indexed list:
 |      (0, seq[0]), (1, seq[1]), (2, seq[2]), ...
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __next__(self, /)
 |      Implement next(self).
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |  
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |  
 |  __class_getitem__(...)
 |      See PEP 585
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.

class filter(object)
 |  filter(function or None, iterable) --> filter object
 |  
 |  Return an iterator yielding those items of iterable for which function(item)
 |  is true. If function is None, return the items that are true.
 |  
 |  Methods defined here:
 |  
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |  
 |  __iter__(self, /)
 |      Implement iter(self).
 |  
 |  __next__(self, /)
 |      Implement next(self).
 |  
 |  __reduce__(...)
 |      Return state information for pickling.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.

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.
 |  
 |  __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 a pair of integers, whose ratio is exactly equal to the original float.
 |
 |      The ratio is in lowest terms and has 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, /)
 |      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 'IEEE,
 |      little-endian' best describes the format of floating-point numbers used by the
 |      C type named by typestr.
 |
 |  fromhex(string, /)
 |      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)
     |      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
    
    class frozenset(object)
     |  frozenset() -> empty frozenset object
     |  frozenset(iterable) -> frozenset object
     |
     |  Build an immutable unordered collection of unique elements.
     |
     |  Methods defined here:
     |
     |  __and__(self, value, /)
     |      Return self&value.
     |
     |  __contains__(...)
     |      x.__contains__(y) <==> y in x.
     |
     |  __eq__(self, value, /)
     |      Return self==value.
     |
     |  __ge__(self, value, /)
     |      Return self>=value.
     |
     |  __gt__(self, value, /)
     |      Return self>value.
     |
     |  __hash__(self, /)
     |      Return hash(self).
     |
     |  __iter__(self, /)
     |      Implement iter(self).
     |
     |  __le__(self, value, /)
     |      Return self<=value.
     |
     |  __len__(self, /)
     |      Return len(self).
     |
     |  __lt__(self, value, /)
     |      Return self<value.
     |
     |  __ne__(self, value, /)
     |      Return self!=value.
     |
     |  __or__(self, value, /)
     |      Return self|value.
     |
     |  __rand__(self, value, /)
     |      Return value&self.
     |
     |  __reduce__(...)
     |      Return state information for pickling.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __ror__(self, value, /)
     |      Return value|self.
     |
     |  __rsub__(self, value, /)
     |      Return value-self.
     |
     |  __rxor__(self, value, /)
     |      Return value^self.
     |
     |  __sizeof__(...)
     |      S.__sizeof__() -> size of S in memory, in bytes
     |
     |  __sub__(self, value, /)
     |      Return self-value.
     |
     |  __xor__(self, value, /)
     |      Return self^value.
     |
     |  copy(...)
     |      Return a shallow copy of a set.
     |
     |  difference(...)
     |      Return the difference of two or more sets as a new set.
     |
     |      (i.e. all elements that are in this set but not the others.)
     |
     |  intersection(...)
     |      Return the intersection of two sets as a new set.
     |
     |      (i.e. all elements that are in both sets.)
     |
     |  isdisjoint(...)
     |      Return True if two sets have a null intersection.
```

```
 |
 |  issubset(self, other, /)
 |      Test whether every element in the set is in other.
 |
 |  issuperset(self, other, /)
 |      Test whether every element in other is in the set.
 |
 |  symmetric_difference(...)
 |      Return the symmetric difference of two sets as a new set.
 |
 |      (i.e. all elements that are in exactly one of the sets.)
 |
 |  union(...)
 |      Return the union of sets as a new set.
 |
 |      (i.e. all elements that are in either set.)
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.

class int(object)
 |  int([x]) -> integer
 |  int(x, base=10) -> integer
 |
 |  Convert a number or string to an integer, or return 0 if no arguments
 |  are given.  If x is a number, return x.__int__().  For floating-point
 |  numbers, this truncates towards zero.
 |
 |  If x is not a number or if base is given, then x must be a string,
 |  bytes, or bytearray instance representing an integer literal in the
 |  given base.  The literal can be preceded by '+' or '-' and be surrounded
 |  by whitespace.  The base defaults to 10.  Valid bases are 0 and 2-36.
 |  Base 0 means to interpret the base from the string as an integer literal.
 |  >>> int('0b100', base=0)
 |  4
 |
 |  Built-in subclasses:
 |      bool
 |
 |  Methods defined here:
 |
 |  __abs__(self, /)
 |      abs(self)
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __and__(self, value, /)
 |      Return self&value.
 |
 |  __bool__(self, /)
 |      True if self else False
 |
 |  __ceil__(...)
 |      Ceiling of an Integral returns itself.
 |
 |  __divmod__(self, value, /)
 |      Return divmod(self, value).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __float__(self, /)
 |      float(self)
 |
 |  __floor__(...)
 |      Flooring an Integral returns itself.
 |
 |  __floordiv__(self, value, /)
 |      Return self//value.
 |
 |  __format__(self, format_spec, /)
 |      Convert to a string 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).
 |
 |  __index__(self, /)
 |      Return self converted to an integer, if self is suitable for use as an index into a list.
 |
 |  __int__(self, /)
 |      int(self)
 |
 |  __invert__(self, /)
 |      ~self
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __lshift__(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
 |
 |  __or__(self, value, /)
 |      Return self|value.
 |
 |  __pos__(self, /)
 |      +self
 |
 |  __pow__(self, value, mod=None, /)
 |      Return pow(self, value, mod).
 |
 |  __radd__(self, value, /)
 |      Return value+self.
 |
 |  __rand__(self, value, /)
 |      Return value&self.
 |
 |  __rdivmod__(self, value, /)
 |      Return divmod(value, self).
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rfloordiv__(self, value, /)
 |      Return value//self.
 |
 |  __rlshift__(self, value, /)
 |      Return value<<self.
 |
 |  __rmod__(self, value, /)
 |      Return value%self.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __ror__(self, value, /)
 |      Return value|self.
 |
 |  __round__(...)
 |      Rounding an Integral returns itself.
 |
 |      Rounding with an ndigits argument also returns an integer.
 |
 |  __rpow__(self, value, mod=None, /)
 |      Return pow(value, self, mod).
 |
 |  __rrshift__(self, value, /)
 |      Return value>>self.
 |
 |  __rshift__(self, value, /)
 |      Return self>>value.
 |
 |  __rsub__(self, value, /)
 |      Return value-self.
 |
 |  __rtruediv__(self, value, /)
 |      Return value/self.
 |
 |  __rxor__(self, value, /)
 |      Return value^self.
 |
 |  __sizeof__(self, /)
 |      Returns size in memory, in bytes.
 |
 |  __sub__(self, value, /)
 |      Return self-value.
 |
 |  __truediv__(self, value, /)
 |      Return self/value.
 |
 |  __trunc__(...)
 |      Truncating an Integral returns itself.
 |
 |  __xor__(self, value, /)
 |      Return self^value.
 |
 |  as_integer_ratio(self, /)
 |      Return a pair of integers, whose ratio is equal to the original int.
 |
 |      The ratio is in lowest terms and has a positive denominator.
 |
 |      >>> (10).as_integer_ratio()
 |      (10, 1)
 |      >>> (-10).as_integer_ratio()
 |      (-10, 1)
 |      >>> (0).as_integer_ratio()
 |      (0, 1)
 |
 |  bit_count(self, /)
 |      Number of ones in the binary representation of the absolute value of self.
 |
 |      Also known as the population count.
 |
 |      >>> bin(13)
 |      '0b1101'
 |      >>> (13).bit_count()
 |      3
 |
 |  bit_length(self, /)
 |      Number of bits necessary to represent self in binary.
 |
 |      >>> bin(37)
 |      '0b100101'
 |      >>> (37).bit_length()
 |      6
 |
 |  conjugate(...)
 |      Returns self, the complex conjugate of any int.
 |
 |  is_integer(self, /)
 |      Returns True. Exists for duck type compatibility with float.is_integer.
 |
 |  to_bytes(self, /, length=1, byteorder='big', *, signed=False)
 |      Return an array of bytes representing an integer.
 |
 |      length
 |        Length of bytes object to use.  An OverflowError is raised if the
 |        integer is not representable with the given number of bytes.  Default
 |        is length 1.
 |      byteorder
 |        The byte order used to represent the integer.  If byteorder is 'big',
 |        the most significant byte is at the beginning of the byte array.  If
 |        byteorder is 'little', the most significant byte is at the end of the
 |        byte array.  To request the native byte order of the host system, use
 |        `sys.byteorder' as the byte order value.  Default is to use 'big'.
 |      signed
 |        Determines whether two's complement is used to represent the integer.
 |        If signed is False and a negative integer is given, an OverflowError
 |        is raised.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  from_bytes(bytes, byteorder='big', *, signed=False)
 |      Return the integer represented by the given array of bytes.
 |
 |      bytes
 |        Holds the array of bytes to convert.  The argument must either
 |        support the buffer protocol or be an iterable object producing bytes.
 |        Bytes and bytearray are examples of built-in objects that support the
 |        buffer protocol.
 |      byteorder
 |        The byte order used to represent the integer.  If byteorder is 'big',
 |        the most significant byte is at the beginning of the byte array.  If
 |        byteorder is 'little', the most significant byte is at the end of the
 |        byte array.  To request the native byte order of the host system, use
 |        `sys.byteorder' as the byte order value.  Default is to use 'big'.
 |      signed
 |        Indicates whether two's complement is used to represent the integer.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  denominator
 |      the denominator of a rational number in lowest terms
 |
 |  imag
 |      the imaginary part of a complex number
 |
 |  numerator
 |      the numerator of a rational number in lowest terms
 |
 |  real
 |      the real part of a complex number

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 bool(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__(self, index, /)
 |      Return self[index].
 |
 |  __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 them,
 |      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__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None

class map(object)
 |  map(func, *iterables) --> map object
 |
 |  Make an iterator that computes the function using arguments from
 |  each of the iterables.  Stops when the shortest iterable is exhausted.
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __next__(self, /)
 |      Implement next(self).
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.

class memoryview(object)
 |  memoryview(object)
 |
 |  Create a new memoryview object which references the given object.
 |
 |  Methods defined here:
 |
 |  __buffer__(self, flags, /)
 |      Return a buffer object that exposes the underlying memory of the object.
 |
 |  __delitem__(self, key, /)
 |      Delete self[key].
 |
 |  __enter__(...)
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __exit__(...)
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |  
 |  __release_buffer__(self, buffer, /)
 |      Release the buffer object that exposes the underlying memory of the object.
 |  
 |  __repr__(self, /)
 |      Return repr(self).
 |  
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |  
 |  cast(...)
 |      Cast a memoryview to a new format or shape.
 |  
 |  hex(...)
 |      Return the data in the buffer as a str of hexadecimal numbers.
 |      
 |        sep
 |          An optional single character or byte to separate hex bytes.
 |        bytes_per_sep
 |          How many bytes between separators.  Positive values count from the
 |          right, negative values count from the left.
 |      
 |      Example:
 |      >>> value = memoryview(b'\xb9\x01\xef')
 |      >>> value.hex()
 |      'b901ef'
 |      >>> value.hex(':')
 |      'b9:01:ef'
 |      >>> value.hex(':', 2)
 |      'b9:01ef'
 |      >>> value.hex(':', -2)
 |      'b901:ef'
 |  
 |  release(self, /)
 |      Release the underlying buffer exposed by the memoryview object.
 |  
 |  tobytes(self, /, order='C')
 |      Return the data in the buffer as a byte string.
 |      
 |      Order can be {'C', 'F', 'A'}. When order is 'C' or 'F', the data of the
 |      original array is converted to C or Fortran order. For contiguous views,
 |      'A' returns an exact copy of the physical memory. In particular, in-memory
 |      Fortran order is preserved. For non-contiguous views, the data is converted
 |      to C first. order=None is the same as order='C'.
 |  
 |  tolist(self, /)
 |      Return the data in the buffer as a list of elements.
 |  
 |  toreadonly(self, /)
 |      Return a readonly version of the memoryview.
 |  
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |  
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |  
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |  
 |  c_contiguous
 |      A bool indicating whether the memory is C contiguous.
 |  
 |  contiguous
 |      A bool indicating whether the memory is contiguous.
 |  
 |  f_contiguous
 |      A bool indicating whether the memory is Fortran contiguous.
 |  
 |  format
 |      A string containing the format (in struct module style)
 |      for each element in the view.
 |  
 |  itemsize
 |      The size in bytes of each element of the memoryview.
 |  
 |  nbytes
 |      The amount of space in bytes that the array would use in
 |      a contiguous representation.
 |  
 |  ndim
 |      An integer indicating how many dimensions of a multi-dimensional
 |      array the memory represents.
 |  
 |  obj
 |      The underlying object of the memoryview.
 |  
 |  readonly
 |      A bool indicating whether the memory is read only.
 |  
 |  shape
 |      A tuple of ndim integers giving the shape of the memory
 |      as an N-dimensional array.
 |  
 |  strides
 |      A tuple of ndim integers giving the size in bytes to access
 |      each element for each dimension of the array.
 |  
 |  suboffsets
 |      A tuple of integers used internally for PIL-style arrays.
```

```
class object
 |  The base class of the class hierarchy.
 |
 |  When called, it accepts no arguments and returns a new featureless
 |  instance that has no instance attributes and cannot be given any.
 |
 |  Built-in subclasses:
 |      anext_awaitable
 |      async_generator
 |      async_generator_asend
 |      async_generator_athrow
 |      ... and 90 other subclasses
 |
 |  Methods defined here:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __dir__(self, /)
 |      Default dir() implementation.
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __format__(self, format_spec, /)
 |      Default object formatter.
 |
 |      Return str(self) if format_spec is empty. Raise TypeError otherwise.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getstate__(self, /)
 |      Helper for pickle.
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __reduce__(self, /)
 |      Helper for pickle.
 |
 |  __reduce_ex__(self, protocol, /)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __sizeof__(self, /)
 |      Size of object in memory, in bytes.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __init_subclass__(...)
 |      This method is called when a class is subclassed.
 |
 |      The default implementation does nothing. It may be
 |      overridden to extend subclasses.
 |
 |  __subclasshook__(...)
 |      Abstract classes can override this to customize issubclass().
 |
 |      This is invoked early on by abc.ABCMeta.__subclasscheck__().
 |      It should return True, False or NotImplemented.  If it returns
 |      NotImplemented, the normal algorithm is used.  Otherwise, it
 |      overrides the normal algorithm (and the outcome is cached).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __class__ = <class 'type'>
 |      type(object) -> the object's type
 |      type(name, bases, dict, **kwds) -> a new type

class property(object)
 |  property(fget=None, fset=None, fdel=None, doc=None)
 |
 |  Property attribute.
 |
 |    fget
 |      function to be used for getting an attribute value
 |    fset
 |      function to be used for setting an attribute value
 |    fdel
 |      function to be used for del'ing an attribute
 |    doc
 |      docstring
 |
 |  Typical use is to define a managed attribute x:
 |
 |  class C(object):
 |      def getx(self): return self._x
 |      def setx(self, value): self._x = value
 |      def delx(self): del self._x
 |      x = property(getx, setx, delx, "I'm the 'x' property.")
 |
 |  Decorators make defining new properties or modifying existing ones easy:
 |
 |  class C(object):
 |      @property
 |      def x(self):
 |          "I am the 'x' property."
 |          return self._x
 |      @x.setter
 |      def x(self, value):
 |          self._x = value
 |      @x.deleter
 |      def x(self):
 |          del self._x
 |
 |  Methods defined here:
 |
 |  __delete__(self, instance, /)
 |      Delete an attribute of instance.
 |
 |  __get__(self, instance, owner=None, /)
 |      Return an attribute of instance, which is of type owner.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __set__(self, instance, value, /)
 |      Set an attribute of instance to value.
 |
 |  __set_name__(...)
 |      Method to set name of a property.
 |
 |  deleter(...)
 |      Descriptor to obtain a copy of the property with a different deleter.
 |
 |  getter(...)
 |      Descriptor to obtain a copy of the property with a different getter.
 |
 |  setter(...)
 |      Descriptor to obtain a copy of the property with a different setter.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __isabstractmethod__
 |
 |  fdel
 |
 |  fget
 |
 |  fset

class range(object)
 |  range(stop) -> range object
 |  range(start, stop[, step]) -> range object
 |
 |  Return an object that produces a sequence of integers from start (inclusive)
 |  to stop (exclusive) by step.  range(i, j) produces i, i+1, i+2, ..., j-1.
 |  start defaults to 0, and stop is omitted!  range(4) produces 0, 1, 2, 3.
 |  These are exactly the valid indices for a list of 4 elements.
 |  When step is given, it specifies the increment (or decrement).
 |
 |  Methods defined here:
 |
 |  __bool__(self, /)
 |      True if self else False
 |
 |  __contains__(self, key, /)
 |      Return bool(key in self).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __reduce__(...)
 |      Helper for pickle.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __reversed__(...)
 |      Return a reverse iterator.
 |
 |  count(...)
 |      rangeobject.count(value) -> integer -- return number of occurrences of value
 |
 |  index(...)
 |      rangeobject.index(value) -> integer -- return index of value.
 |      Raise ValueError if the value is not present.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  start
 |
 |  step
 |
 |  stop

class reversed(object)
 |  reversed(sequence, /)
 |
 |  Return a reverse iterator over the values of the given sequence.
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __length_hint__(...)
 |      Private method returning an estimate of len(list(it)).
 |
 |  __next__(self, /)
 |      Implement next(self).
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  __setstate__(...)
 |      Set state information for unpickling.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.

class set(object)
 |  set() -> new empty set object
 |  set(iterable) -> new set object
 |
 |  Build an unordered collection of unique elements.
 |
 |  Methods defined here:
```

```
 |
 |  __and__(self, value, /)
 |      Return self&value.
 |
 |  __contains__(...)
 |      x.__contains__(y) <==> y in x.
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __iand__(self, value, /)
 |      Return self&=value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __ior__(self, value, /)
 |      Return self|=value.
 |
 |  __isub__(self, value, /)
 |      Return self-=value.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __ixor__(self, value, /)
 |      Return self^=value.
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __or__(self, value, /)
 |      Return self|value.
 |
 |  __rand__(self, value, /)
 |      Return value&self.
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __ror__(self, value, /)
 |      Return value|self.
 |
 |  __rsub__(self, value, /)
 |      Return value-self.
 |
 |  __rxor__(self, value, /)
 |      Return value^self.
 |
 |  __sizeof__(...)
 |      S.__sizeof__() -> size of S in memory, in bytes
 |
 |  __sub__(self, value, /)
 |      Return self-value.
 |
 |  __xor__(self, value, /)
 |      Return self^value.
 |
 |  add(...)
 |      Add an element to a set.
 |
 |      This has no effect if the element is already present.
 |
 |  clear(...)
 |      Remove all elements from this set.
 |
 |  copy(...)
 |      Return a shallow copy of a set.
 |
 |  difference(...)
 |      Return the difference of two or more sets as a new set.
 |
 |      (i.e. all elements that are in this set but not the others.)
 |
 |  difference_update(...)
 |      Remove all elements of another set from this set.
 |
 |  discard(...)
 |      Remove an element from a set if it is a member.
 |
 |      Unlike set.remove(), the discard() method does not raise
 |      an exception when an element is missing from the set.
 |
 |  intersection(...)
 |      Return the intersection of two sets as a new set.
 |
 |      (i.e. all elements that are in both sets.)
 |
 |  intersection_update(...)
 |      Update a set with the intersection of itself and another.
 |
 |  isdisjoint(...)
 |      Return True if two sets have a null intersection.
 |
 |  issubset(self, other, /)
 |      Test whether every element in the set is in other.
 |
 |  issuperset(self, other, /)
 |      Test whether every element in other is in the set.
 |
 |  pop(...)
 |      Remove and return an arbitrary set element.
 |      Raises KeyError if the set is empty.
 |
 |  remove(...)
 |      Remove an element from a set; it must be a member.
 |
 |      If the element is not a member, raise a KeyError.
 |
 |  symmetric_difference(...)
 |      Return the symmetric difference of two sets as a new set.
 |
 |      (i.e. all elements that are in exactly one of the sets.)
 |
 |  symmetric_difference_update(...)
 |      Update a set with the symmetric difference of itself and another.
 |
 |  union(...)
 |      Return the union of sets as a new set.
 |
 |      (i.e. all elements that are in either set.)
 |
 |  update(...)
 |      Update a set with the union of itself and others.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None

class slice(object)
 |  slice(stop)
 |  slice(start, stop[, step])
 |
 |  Create a slice object.  This is used for extended slicing (e.g. a[0:10:2]).
 |
 |  Methods defined here:
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  indices(...)
 |      S.indices(len) -> (start, stop, stride)
 |
 |      Assuming a sequence of length len, calculate the start and stop
 |      indices, and the stride length of the extended slice described by
 |      S. Out of bounds indices are clipped in a manner consistent with the
 |      handling of normal slices.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  start
 |
 |  step
 |
 |  stop

class staticmethod(object)
 |  staticmethod(function) -> method
 |
 |  Convert a function to be a static method.
 |
 |  A static method does not receive an implicit first argument.
 |  To declare a static method, use this idiom:
 |
 |       class C:
 |           @staticmethod
 |           def f(arg1, arg2, argN):
 |               ...
 |
 |  It can be called either on the class (e.g. C.f()) or on an instance
 |  (e.g. C().f()). Both the class and the instance are ignored, and
 |  neither is passed implicitly as the first argument to the method.
 |
 |  Static methods in Python are similar to those found in Java or C++.
 |  For a more advanced concept, see the classmethod builtin.
 |
 |  Methods defined here:
 |
 |  __call__(self, /, *args, **kwargs)
 |      Call self as a function.
 |
 |  __get__(self, instance, owner=None, /)
 |      Return an attribute of instance, which is of type owner.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |
 |  __func__
 |
 |  __isabstractmethod__
 |
 |  __wrapped__

class str(object)
 |  str(object='') -> str
 |  str(bytes_or_buffer[, encoding[, errors]]) -> str
 |
 |  Create a new string object from the given object. If encoding or
 |  errors is specified, then the object must expose a data buffer
 |  that will be decoded using the given encoding and error handler.
 |  Otherwise, returns the result of object.__str__() (if defined)
 |  or repr(object).
 |  encoding defaults to sys.getdefaultencoding().
 |  errors defaults to 'strict'.
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return bool(key in self).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __format__(self, format_spec, /)
 |      Return a formatted version of the string as described by format_spec.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
```

```
 |
 |  __getnewargs__(...)
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __mod__(self, value, /)
 |      Return self%value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rmod__(self, value, /)
 |      Return value%self.
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  __sizeof__(self, /)
 |      Return the size of the string in memory, in bytes.
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  capitalize(self, /)
 |      Return a capitalized version of the string.
 |
 |      More specifically, make the first character have upper case and the rest lower
 |      case.
 |
 |  casefold(self, /)
 |      Return a version of the string suitable for caseless comparisons.
 |
 |  center(self, width, fillchar=' ', /)
 |      Return a centered string of length width.
 |
 |      Padding is done using the specified fill character (default is a space).
 |
 |  count(...)
 |      S.count(sub[, start[, end]]) -> int
 |
 |      Return the number of non-overlapping occurrences of substring sub in
 |      string S[start:end].  Optional arguments start and end are
 |      interpreted as in slice notation.
 |
 |  encode(self, /, encoding='utf-8', errors='strict')
 |      Encode the string using the codec registered for encoding.
 |
 |      encoding
 |        The encoding in which to encode the string.
 |      errors
 |        The error handling scheme to use for encoding errors.
 |        The default is 'strict' meaning that encoding errors raise a
 |        UnicodeEncodeError.  Other possible values are 'ignore', 'replace' and
 |        'xmlcharrefreplace' as well as any other name registered with
 |        codecs.register_error that can handle UnicodeEncodeErrors.
 |
 |  endswith(...)
 |      S.endswith(suffix[, start[, end]]) -> bool
 |
 |      Return True if S ends with the specified suffix, False otherwise.
 |      With optional start, test S beginning at that position.
 |      With optional end, stop comparing S at that position.
 |      suffix can also be a tuple of strings to try.
 |
 |  expandtabs(self, /, tabsize=8)
 |      Return a copy where all tab characters are expanded using spaces.
 |
 |      If tabsize is not given, a tab size of 8 characters is assumed.
 |
 |  find(...)
 |      S.find(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in S where substring sub is found,
 |      such that sub is contained within S[start:end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Return -1 on failure.
 |
 |  format(...)
 |      S.format(*args, **kwargs) -> str
 |
 |      Return a formatted version of S, using substitutions from args and kwargs.
 |      The substitutions are identified by braces ('{' and '}').
 |
 |  format_map(...)
 |      S.format_map(mapping) -> str
 |
 |      Return a formatted version of S, using substitutions from mapping.
 |      The substitutions are identified by braces ('{' and '}').
 |
 |  index(...)
 |      S.index(sub[, start[, end]]) -> int
 |
 |      Return the lowest index in S where substring sub is found,
 |      such that sub is contained within S[start:end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |
 |      Raises ValueError when the substring is not found.
 |
 |  isalnum(self, /)
 |      Return True if the string is an alpha-numeric string, False otherwise.
 |
 |      A string is alpha-numeric if all characters in the string are alpha-numeric and
 |      there is at least one character in the string.
 |
 |  isalpha(self, /)
 |      Return True if the string is an alphabetic string, False otherwise.
 |
 |      A string is alphabetic if all characters in the string are alphabetic and there
 |      is at least one character in the string.
 |
 |  isascii(self, /)
 |      Return True if all characters in the string are ASCII, False otherwise.
 |
 |      ASCII characters have code points in the range U+0000-U+007F.
 |      Empty string is ASCII too.
 |
 |  isdecimal(self, /)
 |      Return True if the string is a decimal string, False otherwise.
 |
 |      A string is a decimal string if all characters in the string are decimal and
 |      there is at least one character in the string.
 |
 |  isdigit(self, /)
 |      Return True if the string is a digit string, False otherwise.
 |
 |      A string is a digit string if all characters in the string are digits and there
 |      is at least one character in the string.
 |
 |  isidentifier(self, /)
 |      Return True if the string is a valid Python identifier, False otherwise.
 |
 |      Call keyword.iskeyword(s) to test whether string s is a reserved identifier,
 |      such as "def" or "class".
 |
 |  islower(self, /)
 |      Return True if the string is a lowercase string, False otherwise.
 |
 |      A string is lowercase if all cased characters in the string are lowercase and
 |      there is at least one cased character in the string.
 |
 |  isnumeric(self, /)
 |      Return True if the string is a numeric string, False otherwise.
 |
 |      A string is numeric if all characters in the string are numeric and there is at
 |      least one character in the string.
 |
 |  isprintable(self, /)
 |      Return True if the string is printable, False otherwise.
 |
 |      A string is printable if all of its characters are considered printable in
 |      repr() or if it is empty.
 |
 |  isspace(self, /)
 |      Return True if the string is a whitespace string, False otherwise.
 |
 |      A string is whitespace if all characters in the string are whitespace and there
 |      is at least one character in the string.
 |
 |  istitle(self, /)
 |      Return True if the string is a title-cased string, False otherwise.
 |
 |      In a title-cased string, upper- and title-case characters may only
 |      follow uncased characters and lowercase characters only cased ones.
 |
 |  isupper(self, /)
 |      Return True if the string is an uppercase string, False otherwise.
 |
 |      A string is uppercase if all cased characters in the string are uppercase and
 |      there is at least one cased character in the string.
 |
 |  join(self, iterable, /)
 |      Concatenate any number of strings.
 |
 |      The string whose method is called is inserted in between each given string.
 |      The result is returned as a new string.
 |
 |      Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'
 |
 |  ljust(self, width, fillchar=' ', /)
 |      Return a left-justified string of length width.
 |
 |      Padding is done using the specified fill character (default is a space).
 |  
 |  lower(self, /)
 |      Return a copy of the string converted to lowercase.
 |  
 |  lstrip(self, chars=None, /)
 |      Return a copy of the string with leading whitespace removed.
 |  
 |      If chars is given and not None, remove characters in chars instead.
 |  
 |  partition(self, sep, /)
 |      Partition the string into three parts using the given separator.
 |  
 |      This will search for the separator in the string.  If the separator is found,
 |      returns a 3-tuple containing the part before the separator, the separator
 |      itself, and the part after it.
 |  
 |      If the separator is not found, returns a 3-tuple containing the original string
 |      and two empty strings.
 |  
 |  removeprefix(self, prefix, /)
 |      Return a str with the given prefix string removed if present.
 |  
 |      If the string starts with the prefix string, return string[len(prefix):].
 |      Otherwise, return a copy of the original string.
 |  
 |  removesuffix(self, suffix, /)
 |      Return a str with the given suffix string removed if present.
 |  
 |      If the string ends with the suffix string and that suffix is not empty,
 |      return string[:-len(suffix)]. Otherwise, return a copy of the original
 |      string.
 |  
 |  replace(self, old, new, count=-1, /)
 |      Return a copy with all occurrences of substring old replaced by new.
 |  
 |        count
 |          Maximum number of occurrences to replace.
 |          -1 (the default value) means replace all occurrences.
 |  
 |      If the optional argument count is given, only the first count occurrences are
 |      replaced.
 |  
 |  rfind(...)
 |      S.rfind(sub[, start[, end]]) -> int
 |  
 |      Return the highest index in S where substring sub is found,
 |      such that sub is contained within S[start:end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |  
 |      Return -1 on failure.
 |  
 |  rindex(...)
 |      S.rindex(sub[, start[, end]]) -> int
 |  
 |      Return the highest index in S where substring sub is found,
 |      such that sub is contained within S[start:end].  Optional
 |      arguments start and end are interpreted as in slice notation.
 |  
 |      Raises ValueError when the substring is not found.
 |  
 |  rjust(self, width, fillchar=' ', /)
 |      Return a right-justified string of length width.
 |  
 |      Padding is done using the specified fill character (default is a space).
 |  
 |  rpartition(self, sep, /)
 |      Partition the string into three parts using the given separator.
 |  
 |      This will search for the separator in the string, starting at the end. If
 |      the separator is found, returns a 3-tuple containing the part before the
 |      separator, the separator itself, and the part after it.
 |  
 |      If the separator is not found, returns a 3-tuple containing two empty strings
 |      and the original string.
 |  
 |  rsplit(self, /, sep=None, maxsplit=-1)
 |      Return a list of the substrings in the string, using sep as the separator string.
 |  
 |        sep
 |          The separator used to split the string.
 |  
 |          When set to None (the default value), will split on any whitespace
 |          character (including \n \r \t \f and spaces) and will discard
 |          empty strings from the result.
 |        maxsplit
 |          Maximum number of splits.
 |          -1 (the default value) means no limit.
 |  
 |      Splitting starts at the end of the string and works to the front.
 |  
 |  rstrip(self, chars=None, /)
 |      Return a copy of the string with trailing whitespace removed.
 |  
 |      If chars is given and not None, remove characters in chars instead.
 |  
 |  split(self, /, sep=None, maxsplit=-1)
 |      Return a list of the substrings in the string, using sep as the separator string.
 |  
 |        sep
 |          The separator used to split the string.
 |  
 |          When set to None (the default value), will split on any whitespace
 |          character (including \n \r \t \f and spaces) and will discard
 |          empty strings from the result.
 |        maxsplit
 |          Maximum number of splits.
 |          -1 (the default value) means no limit.
 |
 |      Splitting starts at the front of the string and works to the end.
 |
 |      Note, str.split() is mainly useful for data that has been intentionally
 |      delimited.  With natural text that includes punctuation, consider using
 |      the regular expression module.
 |
 |  splitlines(self, /, keepends=False)
 |      Return a list of the lines in the string, breaking at line boundaries.
 |
 |      Line breaks are not included in the resulting list unless keepends is given and
 |      true.
 |
 |  startswith(...)
 |      S.startswith(prefix[, start[, end]]) -> bool
 |
 |      Return True if S starts with the specified prefix, False otherwise.
 |      With optional start, test S beginning at that position.
 |      With optional end, stop comparing S at that position.
 |      prefix can also be a tuple of strings to try.
 |
 |  strip(self, chars=None, /)
 |      Return a copy of the string with leading and trailing whitespace removed.
 |
 |      If chars is given and not None, remove characters in chars instead.
 |
 |  swapcase(self, /)
 |      Convert uppercase characters to lowercase and lowercase characters to uppercase.
 |
 |  title(self, /)
 |      Return a version of the string where each word is titlecased.
 |
 |      More specifically, words start with uppercased characters and all remaining
 |      cased characters have lower case.
 |
 |  translate(self, table, /)
 |      Replace each character in the string using the given translation table.
 |
 |        table
 |          Translation table, which must be a mapping of Unicode ordinals to
 |          Unicode ordinals, strings, or None.
 |
 |      The table must implement lookup/indexing via __getitem__, for instance a
 |      dictionary or list.  If this operation raises LookupError, the character is
 |      left untouched.  Characters mapped to None are deleted.
 |
 |  upper(self, /)
 |      Return a copy of the string converted to uppercase.
 |
 |  zfill(self, width, /)
 |      Pad a numeric string with zeros on the left, to fill a field of the given width.
 |
 |      The string is never truncated.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  maketrans(...)
 |      Return a translation table usable for str.translate().
 |
 |      If there is only one argument, it must be a dictionary mapping Unicode
 |      ordinals (integers) or characters to Unicode ordinals, strings or None.
 |      Character keys will be then converted to ordinals.
 |      If there are two arguments, they must be strings of equal length, and
 |      in the resulting dictionary, each character in x will be mapped to the
 |      character at the same position in y. If there is a third argument, it
 |      must be a string, whose characters will be mapped to None in the result.

class super(object)
 |  super() -> same as super(__class__, <first argument>)
 |  super(type) -> unbound super object
 |  super(type, obj) -> bound super object; requires isinstance(obj, type)
 |  super(type, type2) -> bound super object; requires issubclass(type2, type)
 |  Typical use to call a cooperative superclass method:
 |  class C(B):
 |      def meth(self, arg):
 |          super().meth(arg)
 |  This works for class methods too:
 |  class C(B):
 |      @classmethod
 |      def cmeth(cls, arg):
 |          super().cmeth(arg)
 |
 |  Methods defined here:
 |
 |  __get__(self, instance, owner=None, /)
 |      Return an attribute of instance, which is of type owner.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __self__
 |      the instance invoking super(); may be None
 |
 |  __self_class__
 |      the type of the instance invoking super(); may be None
 |
 |  __thisclass__
 |      the class invoking super()

class tuple(object)
 |  tuple(iterable=(), /)
 |
 |  Built-in immutable sequence.
 |
 |  If no argument is given, the constructor returns an empty tuple.
 |  If iterable is specified the tuple is initialized from iterable's items.
 |
 |  If the argument is a tuple, the return value is the same object.
 |
 |  Built-in subclasses:
 |      asyncgen_hooks
 |      MonthDayNano
 |      UnraisableHookArgs
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return bool(key in self).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
 |
 |  __getnewargs__(self, /)
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __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).
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  count(self, value, /)
 |      Return number of occurrences of value.
 |
 |  index(self, value, start=0, stop=9223372036854775807, /)
 |      Return first index of value.
 |
 |      Raises ValueError if the value is not present.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
```

```
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.

    class type(object)
     |  type(object) -> the object's type
     |  type(name, bases, dict, **kwds) -> a new type
     |
     |  Methods defined here:
     |
     |  __call__(self, /, *args, **kwargs)
     |      Call self as a function.
     |
     |  __delattr__(self, name, /)
     |      Implement delattr(self, name).
     |
     |  __dir__(self, /)
     |      Specialized __dir__ implementation for types.
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  __instancecheck__(self, instance, /)
     |      Check if an object is an instance.
     |
     |  __or__(self, value, /)
     |      Return self|value.
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __ror__(self, value, /)
     |      Return value|self.
     |
     |  __setattr__(self, name, value, /)
     |      Implement setattr(self, name, value).
     |
     |  __sizeof__(self, /)
     |      Return memory consumption of the type object.
     |
     |  __subclasscheck__(self, subclass, /)
     |      Check if a class is a subclass.
     |
     |  __subclasses__(self, /)
     |      Return a list of immediate subclasses.
     |
     |  mro(self, /)
     |      Return a type's method resolution order.
     |
     |  ----------------------------------------------------------------------
     |  Class methods defined here:
     |
     |  __prepare__(...)
     |      __prepare__() -> dict
     |      used to create the namespace for the class statement
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors defined here:
     |
     |  __abstractmethods__
     |
     |  __annotations__
     |
     |  __dict__
     |
     |  __text_signature__
     |
     |  ----------------------------------------------------------------------
     |  Data and other attributes defined here:
     |
     |  __base__ = <class 'object'>
     |      The base class of the class hierarchy.
     |
     |      When called, it accepts no arguments and returns a new featureless
     |      instance that has no instance attributes and cannot be given any.
     |
     |
     |  __bases__ = (<class 'object'>,)
     |
     |  __basicsize__ = 920
     |
     |  __dictoffset__ = 264
     |
     |  __flags__ = 2156420354
     |
     |  __itemsize__ = 40
     |
     |  __mro__ = (<class 'type'>, <class 'object'>)
     |
     |  __type_params__ = ()
     |
     |  __weakrefoffset__ = 368

    class zip(object)
     |  zip(*iterables, strict=False) --> Yield tuples until an input is exhausted.
     |
     |     >>> list(zip('abcdefg', range(3), range(4)))
     |     [('a', 0, 0), ('b', 1, 1), ('c', 2, 2)]
     |
     |  The zip object yields n-length tuples, where n is the number of iterables
     |  passed as positional arguments to zip().  The i-th element in every tuple
     |  comes from the i-th iterable argument to zip().  This continues until the
     |  shortest argument is exhausted.
     |
     |  If strict is true and one of the arguments is exhausted before the others,
     |  raise a ValueError.
     |
     |  Methods defined here:
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __iter__(self, /)
     |      Implement iter(self).
     |
     |  __next__(self, /)
     |      Implement next(self).
     |
     |  __reduce__(...)
     |      Return state information for pickling.
     |
     |  __setstate__(...)
     |      Set state information for unpickling.
     |
     |  ----------------------------------------------------------------------
     |  Static methods defined here:
     |
     |  __new__(*args, **kwargs)
     |      Create and return a new object.  See help(type) for accurate signature.

FUNCTIONS
    __build_class__(...)
        __build_class__(func, name, /, *bases, [metaclass], **kwds) -> class

        Internal helper function used by the class statement.

    __import__(name, globals=None, locals=None, fromlist=(), level=0)
        Import a module.

        Because this function is meant for use by the Python
        interpreter and not for general use, it is better to use
        importlib.import_module() to programmatically import a module.

        The globals argument is only used to determine the context;
        they are not modified.  The locals argument is unused.  The fromlist
        should be a list of names to emulate ``from name import ...``, or an
        empty list to emulate ``import name``.
        When importing a module from a package, note that __import__('A.B', ...)
        returns package A when fromlist is empty, but its submodule B when
        fromlist is not empty.  The level argument is used to determine whether to
        perform absolute or relative imports: 0 is absolute, while a positive number
        is the number of parent directories to search relative to the current module.

    abs(x, /)
        Return the absolute value of the argument.

    aiter(async_iterable, /)
        Return an AsyncIterator for an AsyncIterable object.

    all(iterable, /)
        Return True if bool(x) is True for all values x in the iterable.

        If the iterable is empty, return True.

    anext(...)
        async anext(aiterator[, default])

        Return the next item from the async iterator.  If default is given and the async
        iterator is exhausted, it is returned instead of raising StopAsyncIteration.

    any(iterable, /)
        Return True if bool(x) is True for any x in the iterable.

        If the iterable is empty, return False.

    ascii(obj, /)
        Return an ASCII-only representation of an object.

        As repr(), return a string containing a printable representation of an
        object, but escape the non-ASCII characters in the string returned by
        repr() using \\x, \\u or \\U escapes. This generates a string similar
        to that returned by repr() in Python 2.

    bin(number, /)
        Return the binary representation of an integer.

        >>> bin(2796202)
        '0b1010101010101010101010'

    breakpoint(...)
        breakpoint(*args, **kws)
        Call sys.breakpointhook(*args, **kws).  sys.breakpointhook() must accept
        whatever arguments are passed.

        By default, this drops you into the pdb debugger.

    callable(obj, /)
        Return whether the object is callable (i.e., some kind of function).

        Note that classes are callable, as are instances of classes with a
        __call__() method.

    chr(i, /)
        Return a Unicode string of one character with ordinal i; 0 <= i <= 0x10ffff.

    compile(source, filename, mode, flags=0, dont_inherit=False, optimize=-1, *, _feature_version=-1)
        Compile source into a code object that can be executed by exec() or eval().

        The source code may represent a Python module, statement or expression.
        The filename will be used for run-time error messages.
        The mode must be 'exec' to compile a module, 'single' to compile a
        single (interactive) statement, or 'eval' to compile an expression.
        The flags argument, if present, controls which future statements influence
        the compilation of the code.
        The dont_inherit argument, if true, stops the compilation inheriting
        the effects of any future statements in effect in the code calling
        compile; if absent or false these statements do influence the compilation,
        in addition to any features explicitly specified.

    delattr(obj, name, /)
        Deletes the named attribute from the given object.

        delattr(x, 'y') is equivalent to ``del x.y``

    dir(...)
        Show attributes of an object.

        If called without an argument, return the names in the current scope.
        Else, return an alphabetized list of names comprising (some of) the attributes
        of the given object, and of attributes reachable from it.
        If the object supplies a method named __dir__, it will be used; otherwise
        the default dir() logic is used and returns:
          for a module object: the module's attributes.
          for a class object:  its attributes, and recursively the attributes
            of its bases.
          for any other object: its attributes, its class's attributes, and
            recursively the attributes of its class's base classes.

    divmod(x, y, /)
        Return the tuple (x//y, x%y).  Invariant: div*y + mod == x.

    eval(source, globals=None, locals=None, /)
        Evaluate the given source in the context of globals and locals.

        The source may be a string representing a Python expression
        or a code object as returned by compile().
        The globals must be a dictionary and locals can be any mapping,
        defaulting to the current globals and locals.
        If only globals is given, locals defaults to it.

    exec(source, globals=None, locals=None, /, *, closure=None)
        Execute the given source in the context of globals and locals.

        The source may be a string representing one or more Python statements
        or a code object as returned by compile().
        The globals must be a dictionary and locals can be any mapping,
        defaulting to the current globals and locals.
        If only globals is given, locals defaults to it.
        The closure must be a tuple of cellvars, and can only be used
        when source is a code object requiring exactly that many cellvars.

    format(value, format_spec='', /)
        Return type(value).__format__(value, format_spec)

        Many built-in types implement format_spec according to the
        Format Specification Mini-language. See help('FORMATTING').

        If type(value) does not supply a method named __format__
        and format_spec is empty, then str(value) is returned.
        See also help('SPECIALMETHODS').

    getattr(...)
        Get a named attribute from an object.

        getattr(x, 'y') is equivalent to x.y
        When a default argument is given, it is returned when the attribute doesn't
        exist; without it, an exception is raised in that case.

    globals()
        Return the dictionary containing the current scope's global variables.

        NOTE: Updates to this dictionary *will* affect name lookups in the current
        global scope and vice-versa.

    hasattr(obj, name, /)
        Return whether the object has an attribute with the given name.

        This is done by calling getattr(obj, name) and catching AttributeError.

    hash(obj, /)
        Return the hash value for the given object.

        Two objects that compare equal must also have the same hash value, but the
        reverse is not necessarily true.

    hex(number, /)
        Return the hexadecimal representation of an integer.

        >>> hex(12648430)
        '0xc0ffee'

    id(obj, /)
        Return the identity of an object.

        This is guaranteed to be unique among simultaneously existing objects.
        (CPython uses the object's memory address.)

    isinstance(obj, class_or_tuple, /)
        Return whether an object is an instance of a class or of a subclass thereof.

        A tuple, as in ``isinstance(x, (A, B, ...))``, may be given as the target to
        check against. This is equivalent to ``isinstance(x, A) or isinstance(x, B)
        or ...`` etc.

    issubclass(cls, class_or_tuple, /)
        Return whether 'cls' is derived from another class or is the same class.

        A tuple, as in ``issubclass(x, (A, B, ...))``, may be given as the target to
        check against. This is equivalent to ``issubclass(x, A) or issubclass(x, B)
        or ...``.

    iter(...)
        Get an iterator from an object.

        In the first form, the argument must supply its own iterator, or be a sequence.
        In the second form, the callable is called until it returns the sentinel.

    len(obj, /)
        Return the number of items in a container.

    locals()
        Return a dictionary containing the current scope's local variables.

        NOTE: Whether or not updates to this dictionary will affect name lookups in
        the local scope and vice-versa is *implementation dependent* and not
        covered by any backwards compatibility guarantees.

    max(...)
        max(iterable, *[, default=obj, key=func]) -> value
        max(arg1, arg2, *args, *[, key=func]) -> value

        With a single iterable argument, return its biggest item. The
        default keyword-only argument specifies an object to return if
        the provided iterable is empty.
        With two or more arguments, return the largest argument.

    min(...)
        min(iterable, *[, default=obj, key=func]) -> value
        min(arg1, arg2, *args, *[, key=func]) -> value

        With a single iterable argument, return its smallest item. The
        default keyword-only argument specifies an object to return if
        the provided iterable is empty.
        With two or more arguments, return the smallest argument.

    next(...)
        Return the next item from the iterator.

        If default is given and the iterator is exhausted,
        it is returned instead of raising StopIteration.

    oct(number, /)
        Return the octal representation of an integer.

        >>> oct(342391)
        '0o1234567'

    open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)
        Open file and return a stream.  Raise OSError upon failure.

        file is either a text or byte string giving the name (and the path
        if the file isn't in the current working directory) of the file to
        be opened or an integer file descriptor of the file to be
        wrapped. (If a file descriptor is given, it is closed when the
        returned I/O object is closed, unless closefd is set to False.)

        mode is an optional string that specifies the mode in which the file
        is opened. It defaults to 'r' which means open for reading in text
        mode.  Other common values are 'w' for writing (truncating the file if
        it already exists), 'x' for creating and writing to a new file, and
        'a' for appending (which on some Unix systems, means that all writes
        append to the end of the file regardless of the current seek position).
        In text mode, if encoding is not specified the encoding used is platform
        dependent: locale.getencoding() is called to get the current locale encoding.
        (For reading and writing raw bytes use binary mode and leave encoding
        unspecified.) The available modes are:

        ========= ===============================================================
        Character Meaning
        --------- ---------------------------------------------------------------
        'r'       open for reading (default)
        'w'       open for writing, truncating the file first
        'x'       create a new file and open it for writing
        'a'       open for writing, appending to the end of the file if it exists
        'b'       binary mode
        't'       text mode (default)
        '+'       open a disk file for updating (reading and writing)
        ========= ===============================================================

        The default mode is 'rt' (open for reading text). For binary random
        access, the mode 'w+b' opens and truncates the file to 0 bytes, while
        'r+b' opens the file without truncation. The 'x' mode implies 'w' and
        raises an `FileExistsError` if the file already exists.

        Python distinguishes between files opened in binary and text modes,
        even when the underlying operating system doesn't. Files opened in
        binary mode (appending 'b' to the mode argument) return contents as
        bytes objects without any decoding. In text mode (the default, or when
        't' is appended to the mode argument), the contents of the file are
        returned as strings, the bytes having been first decoded using a
        platform-dependent encoding or using the specified encoding if given.

        buffering is an optional integer used to set the buffering policy.
        Pass 0 to switch buffering off (only allowed in binary mode), 1 to select
        line buffering (only usable in text mode), and an integer > 1 to indicate
        the size of a fixed-size chunk buffer.  When no buffering argument is
        given, the default buffering policy works as follows:

        * Binary files are buffered in fixed-size chunks; the size of the buffer
          is chosen using a heuristic trying to determine the underlying device's
          "block size" and falling back on `io.DEFAULT_BUFFER_SIZE`.
          On many systems, the buffer will typically be 4096 or 8192 bytes long.

        * "Interactive" text files (files for which isatty() returns True)
          use line buffering.  Other text files use the policy described above
          for binary files.

        encoding is the name of the encoding used to decode or encode the
        file. This should only be used in text mode. The default encoding is
        platform dependent, but any encoding supported by Python can be
        passed.  See the codecs module for the list of supported encodings.

        errors is an optional string that specifies how encoding errors are to
        be handled---this argument should not be used in binary mode. Pass
        'strict' to raise a ValueError exception if there is an encoding error
        (the default of None has the same effect), or pass 'ignore' to ignore
        errors. (Note that ignoring encoding errors can lead to data loss.)
        See the documentation for codecs.register or run 'help(codecs.Codec)'
        for a list of the permitted encoding error strings.

        newline controls how universal newlines works (it only applies to text
        mode). It can be None, '', '\n', '\r', and '\r\n'.  It works as
        follows:

        * On input, if newline is None, universal newlines mode is
          enabled. Lines in the input can end in '\n', '\r', or '\r\n', and
          these are translated into '\n' before being returned to the
          caller. If it is '', universal newline mode is enabled, but line
          endings are returned to the caller untranslated. If it has any of
          the other legal values, input lines are only terminated by the given
          string, and the line ending is returned to the caller untranslated.

        * On output, if newline is None, any '\n' characters written are
          translated to the system default line separator, os.linesep. If
          newline is '' or '\n', no translation takes place. If newline is any
          of the other legal values, any '\n' characters written are translated
          to the given string.

        If closefd is False, the underlying file descriptor will be kept open
        when the file is closed. This does not work when a file name is given
        and must be True in that case.

        A custom opener can be used by passing a callable as *opener*. The
        underlying file descriptor for the file object is then obtained by
        calling *opener* with (*file*, *flags*). *opener* must return an open
        file descriptor (passing os.open as *opener* results in functionality
        similar to passing None).

        open() returns a file object whose type depends on the mode, and
        through which the standard file operations such as reading and writing
        are performed. When open() is used to open a file in a text mode ('w',
        'r', 'wt', 'rt', etc.), it returns a TextIOWrapper. When used to open
        a file in a binary mode, the returned class varies: in read binary
        mode, it returns a BufferedReader; in write binary and append binary
        modes, it returns a BufferedWriter, and in read/write mode, it returns
        a BufferedRandom.

        It is also possible to use a string or bytearray as a file for both
        reading and writing. For strings StringIO can be used like a file
        opened in a text mode, and for bytes a BytesIO can be used like a file
        opened in a binary mode.

    ord(c, /)
        Return the Unicode code point for a one-character string.

    pow(base, exp, mod=None)
        Equivalent to base**exp with 2 arguments or base**exp % mod with 3 arguments

        Some types, such as ints, are able to use a more efficient algorithm when
        invoked using the three argument form.

    print(*args, sep=' ', end='\n', file=None, flush=False)
        Prints the values to a stream, or to sys.stdout by default.

        sep
          string inserted between values, default a space.
        end
```

```
          string appended after the last value, default a newline.
        file
          a file-like object (stream); defaults to the current sys.stdout.
        flush
          whether to forcibly flush the stream.

    repr(obj, /)
        Return the canonical string representation of the object.

        For many object types, including most builtins, eval(repr(obj)) == obj.

    round(number, ndigits=None)
        Round a number to a given precision in decimal digits.

        The return value is an integer if ndigits is omitted or None.  Otherwise
        the return value has the same type as the number.  ndigits may be negative.

    setattr(obj, name, value, /)
        Sets the named attribute on the given object to the specified value.

        setattr(x, 'y', v) is equivalent to ``x.y = v``

    sorted(iterable, /, *, key=None, reverse=False)
        Return a new list containing all items from the iterable in ascending order.

        A custom key function can be supplied to customize the sort order, and the
        reverse flag can be set to request the result in descending order.

    sum(iterable, /, start=0)
        Return the sum of a 'start' value (default: 0) plus an iterable of numbers

        When the iterable is empty, return the start value.
        This function is intended specifically for use with numeric values and may
        reject non-numeric types.

    vars(...)
        Show vars.

        Without arguments, equivalent to locals().
        With an argument, equivalent to object.__dict__.

DATA
    Ellipsis = Ellipsis
    False = False
    None = None
    NotImplemented = NotImplemented
    True = True
    __IPYTHON__ = True
    __debug__ = True
    copyright = Copyright (c) 2001-2023 Python Software Foundati...ematisc...
    credits =     Thanks to CWI, CNRI, BeOpen.com, Zope Corpor...opment.  ...
    help = Type help() for interactive help, or help(object) for help abou...
    license = See https://www.python.org/psf/license/

FILE
    (built-in)


You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
```

In [674…
```python
num =5
num
```

Out[674…
```
5
```

In [676…
```python
id(num)
```

Out[676…
```
140717641181752
```

In [680…
```python
nums2
```

Out[680…
```
[1, 'hi', 29, 14, 35]
```

range()

In [113…
```python
r = range(0,10)
r
```

Out[113…
```
range(0, 10)
```

type(r)

In [113…
```python
list(range(0,10)) # print th range
r1 = list(r)
r1
```

Out[113…
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [113…
```python
# print even numbers
even_number = list(range(1,8,2))
even_number
```

Out[113…
```
[1, 3, 5, 7]
```

In [113…
```python
d ={'a':'one','b':list(range(0,10))}
d
```

```
Out[113… {'a': 'one', 'b': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]}
```

```
In [113… d[0]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[1136], line 1
----> 1 d[0]

KeyError: 0
```

```
In [114… d['b'][2:5]
```

```
C:\ProgramData\anaconda3\Lib\site-packages\IPython\core\displayhook.py:281: UserWarning: Output cache limit (currently 1000 entries) hit.
Flushing oldest 200 entries.
  warn('Output cache limit (currently {sz} entries) hit.\n'
```

```
Out[114… [2, 3, 4]
```

```
In [114… d.get('a')
```

```
Out[114… 'one'
```

```
In [114… d.get('a','b')
```

```
Out[114… 'one'
```

```
In [114… d
```

```
Out[114… {'a': 'one', 'b': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]}
```

Operator

- Arithmetic operator
- Assignment Operator
- Relational Operator
- Logical Operator
- Unary Operator

Arithmetic Operator

```
In [115… x1, y1 = 2, 3
         print(x1)
         print(y1)
```

```
2
3
```

```
In [115… x1 + y1
```

```
Out[115… 5
```

```
In [115… x1 -y1
```

```
Out[115… -1
```

```
In [115… x1 * y1
```

```
Out[115… 6
```

```
In [115… x1 /y1
```

```
Out[115… 0.6666666666666666
```

```
In [115… x1 //y1
```

```
Out[115… 0
```

```
In [116… x1 %y1
```

```
Out[116… 2
```

```
In [116… x1 ** y1
```

```
Out[116… 8
```

Assignment Operator

```
In [116… x = 2
```

```
In [116… x = x + 2
         x
```

```
Out[116… 4
```

```
In [117… x +=2
         x
```

```
Out[117… 6
```

```
In [117…  x *=2
          x
Out[117…  12

In [117…   x -=2
          x
Out[117…  10

In [117…  x /=2
          x
Out[117…  5.0

In [117…  x //=2

In [118…  x
Out[118…  2.0
```

Unary operator

```
In [119…  n =7
          n
Out[119…  7

In [119…  m =-(n)
          m
Out[119…  -7

In [119…  -n
Out[119…  -7

In [119…  m ==-n
Out[119…  True

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```