:mod:`operator` --- Standard operators as functions

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 1); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 4)

Unknown directive type "module".

```
.. module:: operator
   :synopsis: Functions corresponding to the standard operators.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 7)

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Skip Montanaro <skip@automatrix.com>
```

Source code: :source:`Lib/operator.py`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 9); *backlink*

Unknown interpreted text role "source".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 11)

Unknown directive type "testsetup".

```
.. testsetup::

   import operator
   from operator import itemgetter, iadd
```

The :mod:`operator` module exports a set of efficient functions corresponding to the intrinsic operators of Python. For example, `operator.add(x, y)` is equivalent to the expression `x+y`. Many function names are those used for special methods, without the double underscores. For backward compatibility, many of these have a variant with the double underscores kept. The variants without the double underscores are preferred for clarity.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 18); *backlink*

Unknown interpreted text role "mod".

The functions fall into categories that perform object comparisons, logical operations, mathematical operations and sequence operations.

The object comparison functions are useful for all objects, and are named after the rich comparison operators they support:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 32)

Unknown directive type "function".

```
.. function:: lt(a, b)
              le(a, b)
              eq(a, b)
              ne(a, b)
              ge(a, b)
              gt(a, b)
```

```
                __lt__(a, b)
                __le__(a, b)
                __eq__(a, b)
                __ne__(a, b)
                __ge__(a, b)
                __gt__(a, b)

   Perform "rich comparisons" between *a* and *b*. Specifically, ``lt(a, b)`` is
   equivalent to ``a < b``, ``le(a, b)`` is equivalent to ``a <= b``, ``eq(a,
   b)`` is equivalent to ``a == b``, ``ne(a, b)`` is equivalent to ``a != b``,
   ``gt(a, b)`` is equivalent to ``a > b`` and ``ge(a, b)`` is equivalent to ``a
   >= b``.  Note that these functions can return any value, which may
   or may not be interpretable as a Boolean value.  See
   :ref:`comparisons` for more information about rich comparisons.
```

The logical operations are also generally applicable to all objects, and support truth tests, identity tests, and boolean operations:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 58)

Unknown directive type "function".

```
.. function:: not_(obj)
              __not__(obj)

   Return the outcome of :keyword:`not` *obj*.  (Note that there is no
   :meth:`__not__` method for object instances; only the interpreter core defines
   this operation.  The result is affected by the :meth:`__bool__` and
   :meth:`__len__` methods.)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 67)

Unknown directive type "function".

```
.. function:: truth(obj)

   Return :const:`True` if *obj* is true, and :const:`False` otherwise.  This is
   equivalent to using the :class:`bool` constructor.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 73)

Unknown directive type "function".

```
.. function:: is_(a, b)

   Return ``a is b``.  Tests object identity.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 78)

Unknown directive type "function".

```
.. function:: is_not(a, b)

   Return ``a is not b``.  Tests object identity.
```

The mathematical and bitwise operations are the most numerous:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 86)

Unknown directive type "function".

```
.. function:: abs(obj)
```

```
                __abs__(obj)

   Return the absolute value of *obj*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 92)

Unknown directive type "function".

```
.. function:: add(a, b)
                __add__(a, b)

   Return ``a + b``, for *a* and *b* numbers.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 98)

Unknown directive type "function".

```
.. function:: and_(a, b)
                __and__(a, b)

   Return the bitwise and of *a* and *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 104)

Unknown directive type "function".

```
.. function:: floordiv(a, b)
                __floordiv__(a, b)

   Return ``a // b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 110)

Unknown directive type "function".

```
.. function:: index(a)
                __index__(a)

   Return *a* converted to an integer.  Equivalent to ``a.__index__()``.

   .. versionchanged:: 3.10
      The result always has exact type :class:`int`.  Previously, the result
      could have been an instance of a subclass of ``int``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 120)

Unknown directive type "function".

```
.. function:: inv(obj)
                invert(obj)
                __inv__(obj)
                __invert__(obj)

   Return the bitwise inverse of the number *obj*.  This is equivalent to ``~obj``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 128)

Unknown directive type "function".

```
.. function:: lshift(a, b)
              __lshift__(a, b)

   Return *a* shifted left by *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 134)

Unknown directive type "function".

```
.. function:: mod(a, b)
              __mod__(a, b)

   Return ``a % b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 140)

Unknown directive type "function".

```
.. function:: mul(a, b)
              __mul__(a, b)

   Return ``a * b``, for *a* and *b* numbers.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 146)

Unknown directive type "function".

```
.. function:: matmul(a, b)
              __matmul__(a, b)

   Return ``a @ b``.

   .. versionadded:: 3.5
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 154)

Unknown directive type "function".

```
.. function:: neg(obj)
              __neg__(obj)

   Return *obj* negated (``-obj``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 160)

Unknown directive type "function".

```
.. function:: or_(a, b)
              __or__(a, b)

   Return the bitwise or of *a* and *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 166)

Unknown directive type "function".

```
.. function:: pos(obj)
              __pos__(obj)

   Return *obj* positive (``+obj``).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 172)

Unknown directive type "function".

```
.. function:: pow(a, b)
              __pow__(a, b)

   Return ``a ** b``, for *a* and *b* numbers.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 178)

Unknown directive type "function".

```
.. function:: rshift(a, b)
              __rshift__(a, b)

   Return *a* shifted right by *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 184)

Unknown directive type "function".

```
.. function:: sub(a, b)
              __sub__(a, b)

   Return ``a - b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 190)

Unknown directive type "function".

```
.. function:: truediv(a, b)
              __truediv__(a, b)

   Return ``a / b`` where 2/3 is .66 rather than 0.  This is also known as
   "true" division.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 197)

Unknown directive type "function".

```
.. function:: xor(a, b)
              __xor__(a, b)

   Return the bitwise exclusive or of *a* and *b*.
```

Operations which work with sequences (some of them with mappings too) include:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 205)

Unknown directive type "function".

```
.. function:: concat(a, b)
              __concat__(a, b)

   Return ``a + b`` for *a* and *b* sequences.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 211)

Unknown directive type "function".

```
.. function:: contains(a, b)
              __contains__(a, b)

   Return the outcome of the test ``b in a``. Note the reversed operands.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 217)

Unknown directive type "function".

```
.. function:: countOf(a, b)

   Return the number of occurrences of *b* in *a*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 222)

Unknown directive type "function".

```
.. function:: delitem(a, b)
              __delitem__(a, b)

   Remove the value of *a* at index *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 228)

Unknown directive type "function".

```
.. function:: getitem(a, b)
              __getitem__(a, b)

   Return the value of *a* at index *b*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 234)

Unknown directive type "function".

```
.. function:: indexOf(a, b)

   Return the index of the first of occurrence of *b* in *a*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 239)

Unknown directive type "function".

```
.. function:: setitem(a, b, c)
              __setitem__(a, b, c)

   Set the value of *a* at index *b* to *c*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 245)

Unknown directive type "function".

```
.. function:: length_hint(obj, default=0)

   Return an estimated length for the object *o*. First try to return its
   actual length, then an estimate using :meth:`object.__length_hint__`, and
   finally return the default value.

   .. versionadded:: 3.4
```

The following operation works with callables:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 256)

Unknown directive type "function".

```
.. function:: call(obj, /, *args, **kwargs)
              __call__(obj, /, *args, **kwargs)

   Return ``obj(*args, **kwargs)``.

   .. versionadded:: 3.11
```

The :mod:`operator` module also defines tools for generalized attribute and item lookups. These are useful for making fast field extractors as arguments for :func:`map`, :func:`sorted`, :meth:`itertools.groupby`, or other functions that expect a function argument.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 264); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 264); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 264); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 264); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 270)

Unknown directive type "function".

```
.. function:: attrgetter(attr)
              attrgetter(*attrs)

   Return a callable object that fetches *attr* from its operand.
   If more than one attribute is requested, returns a tuple of attributes.
   The attribute names can also contain dots. For example:

   * After ``f = attrgetter('name')``, the call ``f(b)`` returns ``b.name``.

   * After ``f = attrgetter('name', 'date')``, the call ``f(b)`` returns
     ``(b.name, b.date)``.
```

```
   * After ``f = attrgetter('name.first', 'name.last')``, the call ``f(b)``
     returns ``(b.name.first, b.name.last)``.

   Equivalent to::

      def attrgetter(*items):
          if any(not isinstance(item, str) for item in items):
              raise TypeError('attribute name must be a string')
          if len(items) == 1:
              attr = items[0]
              def g(obj):
                  return resolve_attr(obj, attr)
          else:
              def g(obj):
                  return tuple(resolve_attr(obj, attr) for attr in items)
          return g

      def resolve_attr(obj, attr):
          for name in attr.split("."):
              obj = getattr(obj, name)
          return obj
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 305)

Unknown directive type "function".

```
.. function:: itemgetter(item)
              itemgetter(*items)

   Return a callable object that fetches *item* from its operand using the
   operand's :meth:`__getitem__` method.  If multiple items are specified,
   returns a tuple of lookup values.  For example:

   * After ``f = itemgetter(2)``, the call ``f(r)`` returns ``r[2]``.

   * After ``g = itemgetter(2, 5, 3)``, the call ``g(r)`` returns
     ``(r[2], r[5], r[3])``.

   Equivalent to::

      def itemgetter(*items):
          if len(items) == 1:
              item = items[0]
              def g(obj):
                  return obj[item]
          else:
              def g(obj):
                  return tuple(obj[item] for item in items)
          return g

   The items can be any type accepted by the operand's :meth:`__getitem__`
   method.  Dictionaries accept any hashable value.  Lists, tuples, and
   strings accept an index or a slice:

      >>> itemgetter(1)('ABCDEFG')
      'B'
      >>> itemgetter(1, 3, 5)('ABCDEFG')
      ('B', 'D', 'F')
      >>> itemgetter(slice(2, None))('ABCDEFG')
      'CDEFG'
      >>> soldier = dict(rank='captain', name='dotterbart')
      >>> itemgetter('rank')(soldier)
      'captain'

   Example of using :func:`itemgetter` to retrieve specific fields from a
   tuple record:

      >>> inventory = [('apple', 3), ('banana', 2), ('pear', 5), ('orange', 1)]
      >>> getcount = itemgetter(1)
      >>> list(map(getcount, inventory))
      [3, 2, 5, 1]
      >>> sorted(inventory, key=getcount)
      [('orange', 1), ('banana', 2), ('apple', 3), ('pear', 5)]
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 354)

Unknown directive type "function".

```
.. function:: methodcaller(name, /, *args, **kwargs)

   Return a callable object that calls the method *name* on its operand.  If
   additional arguments and/or keyword arguments are given, they will be given
   to the method as well.  For example:

   * After ``f = methodcaller('name')``, the call ``f(b)`` returns ``b.name()``.

   * After ``f = methodcaller('name', 'foo', bar=1)``, the call ``f(b)``
     returns ``b.name('foo', bar=1)``.

   Equivalent to::

      def methodcaller(name, /, *args, **kwargs):
          def caller(obj):
              return getattr(obj, name)(*args, **kwargs)
          return caller
```

Mapping Operators to Functions

This table shows how abstract operations correspond to operator symbols in the Python syntax and the functions in the :mod:`operator` module.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 378); *backlink*

Unknown interpreted text role "mod".

Operation	Syntax	Function
Addition	`a + b`	`add(a, b)`
Concatenation	`seq1 + seq2`	`concat(seq1, seq2)`
Containment Test	`obj in seq`	`contains(seq, obj)`
Division	`a / b`	`truediv(a, b)`
Division	`a // b`	`floordiv(a, b)`
Bitwise And	`a & b`	`and_(a, b)`
Bitwise Exclusive Or	`a ^ b`	`xor(a, b)`
Bitwise Inversion	`~ a`	`invert(a)`
Bitwise Or	`a | b`	`or_(a, b)`
Exponentiation	`a ** b`	`pow(a, b)`
Identity	`a is b`	`is_(a, b)`
Identity	`a is not b`	`is_not(a, b)`
Indexed Assignment	`obj[k] = v`	`setitem(obj, k, v)`
Indexed Deletion	`del obj[k]`	`delitem(obj, k)`
Indexing	`obj[k]`	`getitem(obj, k)`
Left Shift	`a << b`	`lshift(a, b)`
Modulo	`a % b`	`mod(a, b)`
Multiplication	`a * b`	`mul(a, b)`
Matrix Multiplication	`a @ b`	`matmul(a, b)`
Negation (Arithmetic)	`- a`	`neg(a)`
Negation (Logical)	`not a`	`not_(a)`
Positive	`+ a`	`pos(a)`
Right Shift	`a >> b`	`rshift(a, b)`
Slice Assignment	`seq[i:j] = values`	`setitem(seq, slice(i, j), values)`
Slice Deletion	`del seq[i:j]`	`delitem(seq, slice(i, j))`
Slicing	`seq[i:j]`	`getitem(seq, slice(i, j))`
String Formatting	`s % obj`	`mod(s, obj)`
Subtraction	`a - b`	`sub(a, b)`
Truth Test	`obj`	`truth(obj)`
Ordering	`a < b`	`lt(a, b)`

Operation	Syntax	Function
Ordering	`a <= b`	`le(a, b)`
Equality	`a == b`	`eq(a, b)`
Difference	`a != b`	`ne(a, b)`
Ordering	`a >= b`	`ge(a, b)`
Ordering	`a > b`	`gt(a, b)`

In-place Operators

Many operations have an "in-place" version. Listed below are functions providing a more primitive access to in-place operators than the usual syntax does; for example, the :term:`statement` `x += y` is equivalent to `x = operator.iadd(x, y)`. Another way to put it is to say that `z = operator.iadd(x, y)` is equivalent to the compound statement `z = x; z += y`.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 458); *backlink***
>
> Unknown interpreted text role "term".

In those examples, note that when an in-place method is called, the computation and assignment are performed in two separate steps. The in-place functions listed below only do the first step, calling the in-place method. The second step, assignment, is not handled.

For immutable targets such as strings, numbers, and tuples, the updated value is computed, but not assigned back to the input variable:

```
>>> a = 'hello'
>>> iadd(a, ' world')
'hello world'
>>> a
'hello'
```

For mutable targets such as lists and dictionaries, the in-place method will perform the update, so no subsequent assignment is necessary:

```
>>> s = ['h', 'e', 'l', 'l', 'o']
>>> iadd(s, [' ', 'w', 'o', 'r', 'l', 'd'])
['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']
>>> s
['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']
```

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 488)**
>
> Unknown directive type "function".
>
> ```
> .. function:: iadd(a, b)
>               __iadd__(a, b)
>
>    ``a = iadd(a, b)`` is equivalent to ``a += b``.
> ```

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 494)**
>
> Unknown directive type "function".
>
> ```
> .. function:: iand(a, b)
>               __iand__(a, b)
>
>    ``a = iand(a, b)`` is equivalent to ``a &= b``.
> ```

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 500)**
>
> Unknown directive type "function".
>
> ```
> .. function:: iconcat(a, b)
>               __iconcat__(a, b)
>
>    ``a = iconcat(a, b)`` is equivalent to ``a += b`` for *a* and *b* sequences.
> ```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 506)

Unknown directive type "function".

```
.. function:: ifloordiv(a, b)
              __ifloordiv__(a, b)

   ``a = ifloordiv(a, b)`` is equivalent to ``a //= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 512)

Unknown directive type "function".

```
.. function:: ilshift(a, b)
              __ilshift__(a, b)

   ``a = ilshift(a, b)`` is equivalent to ``a <<= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 518)

Unknown directive type "function".

```
.. function:: imod(a, b)
              __imod__(a, b)

   ``a = imod(a, b)`` is equivalent to ``a %= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 524)

Unknown directive type "function".

```
.. function:: imul(a, b)
              __imul__(a, b)

   ``a = imul(a, b)`` is equivalent to ``a *= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 530)

Unknown directive type "function".

```
.. function:: imatmul(a, b)
              __imatmul__(a, b)

   ``a = imatmul(a, b)`` is equivalent to ``a @= b``.

   .. versionadded:: 3.5
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 538)

Unknown directive type "function".

```
.. function:: ior(a, b)
              __ior__(a, b)

   ``a = ior(a, b)`` is equivalent to ``a |= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 544)

Unknown directive type "function".

```
.. function:: ipow(a, b)
              __ipow__(a, b)

   ``a = ipow(a, b)`` is equivalent to ``a **= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 550)

Unknown directive type "function".

```
.. function:: irshift(a, b)
              __irshift__(a, b)

   ``a = irshift(a, b)`` is equivalent to ``a >>= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 556)

Unknown directive type "function".

```
.. function:: isub(a, b)
              __isub__(a, b)

   ``a = isub(a, b)`` is equivalent to ``a -= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 562)

Unknown directive type "function".

```
.. function:: itruediv(a, b)
              __itruediv__(a, b)

   ``a = itruediv(a, b)`` is equivalent to ``a /= b``.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]operator.rst, line 568)

Unknown directive type "function".

```
.. function:: ixor(a, b)
              __ixor__(a, b)

   ``a = ixor(a, b)`` is equivalent to ``a ^= b``.
```