:mod:`decimal` --- Decimal fixed point and floating point arithmetic

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

Unknown directive type "module".

```
.. module:: decimal
   :synopsis: Implementation of the General Decimal Arithmetic  Specification.
```

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

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Eric Price <eprice at tjhsst.edu>
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 8)

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Facundo Batista <facundo at taniquetil.com.ar>
```

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

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Raymond Hettinger <python at rcn.com>
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 10)

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Aahz <aahz at pobox.com>
```

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

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Tim Peters <tim.one at comcast.net>
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 12)

Unknown directive type "moduleauthor".

```
.. moduleauthor:: Stefan Krah <skrah at bytereef.org>
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 13)

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Raymond D. Hettinger <python at rcn.com>
```

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

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 15); *backlink*

Unknown interpreted text role "source".

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

Unknown directive type "testsetup".

```
.. testsetup:: *

   import decimal
   import math
   from decimal import *
   # make sure each group gets a fresh context
   setcontext(Context())
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 26)

Unknown directive type "testcleanup".

```
.. testcleanup:: *

   # make sure other tests (outside this file) get a fresh context
   setcontext(Context())
```

The :mod:`decimal` module provides support for fast correctly-rounded decimal floating point arithmetic. It offers several advantages over the :class:`float` datatype:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 33); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 33); *backlink*

Unknown interpreted text role "class".

- Decimal "is based on a floating-point model which was designed with people in mind, and necessarily has a paramount guiding principle -- computers must provide an arithmetic that works in the same way as the arithmetic that people learn at school." -- excerpt from the decimal arithmetic specification.
- Decimal numbers can be represented exactly. In contrast, numbers like :const:`1.1` and :const:`2.2` do not have exact representations in binary floating point. End users typically would not expect `1.1 + 2.2` to display as :const:`3.3000000000000003` as it does with binary floating point.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 42); *backlink*

  Unknown interpreted text role "const".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 42); *backlink*

  Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 42**); ***backlink***

Unknown interpreted text role "const".

- The exactness carries over into arithmetic. In decimal floating point, `0.1 + 0.1 + 0.1 - 0.3` is exactly equal to zero. In binary floating point, the result is :const:`5.5511151231257827e-017`. While near to zero, the differences prevent reliable equality testing and differences can accumulate. For this reason, decimal is preferred in accounting applications which have strict equality invariants.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 47**); ***backlink***

Unknown interpreted text role "const".

- The decimal module incorporates a notion of significant places so that `1.30 + 1.20` is :const:`2.50`. The trailing zero is kept to indicate significance. This is the customary presentation for monetary applications. For multiplication, the "schoolbook" approach uses all the figures in the multiplicands. For instance, `1.3 * 1.2` gives :const:`1.56` while `1.30 * 1.20` gives :const:`1.5600`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 54**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 54**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 54**); ***backlink***

Unknown interpreted text role "const".

- Unlike hardware based binary floating point, the decimal module has a user alterable precision (defaulting to 28 places) which can be as large as needed for a given problem:

```
>>> from decimal import *
>>> getcontext().prec = 6
>>> Decimal(1) / Decimal(7)
Decimal('0.142857')
>>> getcontext().prec = 28
>>> Decimal(1) / Decimal(7)
Decimal('0.1428571428571428571428571429')
```

- Both binary and decimal floating point are implemented in terms of published standards. While the built-in float type exposes only a modest portion of its capabilities, the decimal module exposes all required parts of the standard. When needed, the programmer has full control over rounding and signal handling. This includes an option to enforce exact arithmetic by using exceptions to block any inexact operations.
- The decimal module was designed to support "without prejudice, both exact unrounded decimal arithmetic (sometimes called fixed-point arithmetic) and rounded floating-point arithmetic." -- excerpt from the decimal arithmetic specification.

The module design is centered around three concepts: the decimal number, the context for arithmetic, and signals.

A decimal number is immutable. It has a sign, coefficient digits, and an exponent. To preserve significance, the coefficient digits do not truncate trailing zeros. Decimals also include special values such as :const:`Infinity`, :const:`-Infinity`, and :const:`NaN`. The standard also differentiates :const:`-0` from :const:`+0`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 88**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 88); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 88); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 88); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 88); *backlink*

Unknown interpreted text role "const".

The context for arithmetic is an environment specifying precision, rounding rules, limits on exponents, flags indicating the results of operations, and trap enablers which determine whether signals are treated as exceptions. Rounding options include :const:`ROUND_CEILING`, :const:`ROUND_DOWN`, :const:`ROUND_FLOOR`, :const:`ROUND_HALF_DOWN`, :const:`ROUND_HALF_EVEN`, :const:`ROUND_HALF_UP`, :const:`ROUND_UP`, and :const:`ROUND_05UP`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 94); *backlink*

Unknown interpreted text role "const".

Signals are groups of exceptional conditions arising during the course of computation. Depending on the needs of the application, signals may be ignored, considered as informational, or treated as exceptions. The signals in the decimal module are: :const:`Clamped`, :const:`InvalidOperation`, :const:`DivisionByZero`, :const:`Inexact`, :const:`Rounded`, :const:`Subnormal`, :const:`Overflow`, :const:`Underflow` and :const:`FloatOperation`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 101); *backlink*

Unknown interpreted text role "const".

For each signal there is a flag and a trap enabler. When a signal is encountered, its flag is set to one, then, if the trap enabler is set to one, an exception is raised. Flags are sticky, so the user needs to reset them before monitoring a calculation.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 114)

Unknown directive type "seealso".

```
.. seealso::

   * IBM's General Decimal Arithmetic Specification, `The General Decimal Arithmetic
     Specification <http://speleotrove.com/decimal/decarith.html>`_.
```

Quick-start Tutorial

The usual start to using decimals is importing the module, viewing the current context with :func:`getcontext` and, if necessary, setting new values for precision, rounding, or enabled traps:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 127); *backlink*

Unknown interpreted text role "func".

```
>>> from decimal import *
>>> getcontext()
Context(prec=28, rounding=ROUND_HALF_EVEN, Emin=-999999, Emax=999999,
        capitals=1, clamp=0, flags=[], traps=[Overflow, DivisionByZero,
        InvalidOperation])

>>> getcontext().prec = 7       # Set a new precision
```

Decimal instances can be constructed from integers, strings, floats, or tuples. Construction from an integer or a float performs an exact conversion of the value of that integer or float. Decimal numbers include special values such as :const:`NaN` which stands for "Not a number", positive and negative :const:`Infinity`, and :const:`-0`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 139); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 139); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 139); *backlink*

Unknown interpreted text role "const".

```
>>> getcontext().prec = 28
>>> Decimal(10)
Decimal('10')
>>> Decimal('3.14')
Decimal('3.14')
>>> Decimal(3.14)
Decimal('3.140000000000000124344978758017532527446746826171875')
>>> Decimal((0, (3, 1, 4), -2))
Decimal('3.14')
>>> Decimal(str(2.0 ** 0.5))
Decimal('1.4142135623730951')
>>> Decimal(2) ** Decimal('0.5')
Decimal('1.414213562373095048801688724')
>>> Decimal('NaN')
Decimal('NaN')
>>> Decimal('-Infinity')
Decimal('-Infinity')
```

If the :exc:`FloatOperation` signal is trapped, accidental mixing of decimals and floats in constructors or ordering comparisons raises an exception:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 163); *backlink*

Unknown interpreted text role "exc".

```
>>> c = getcontext()
>>> c.traps[FloatOperation] = True
>>> Decimal(3.14)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
decimal.FloatOperation: [<class 'decimal.FloatOperation'>]
>>> Decimal('3.5') < 3.7
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
```

```
decimal.FloatOperation: [<class 'decimal.FloatOperation'>]
>>> Decimal('3.5') == 3.5
True
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 180)

Unknown directive type "versionadded".

```
.. versionadded:: 3.3
```

The significance of a new Decimal is determined solely by the number of digits input. Context precision and rounding only come into play during arithmetic operations.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 186)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   >>> getcontext().prec = 6
   >>> Decimal('3.0')
   Decimal('3.0')
   >>> Decimal('3.1415926535')
   Decimal('3.1415926535')
   >>> Decimal('3.1415926535') + Decimal('2.7182818285')
   Decimal('5.85987')
   >>> getcontext().rounding = ROUND_UP
   >>> Decimal('3.1415926535') + Decimal('2.7182818285')
   Decimal('5.85988')
```

If the internal limits of the C version are exceeded, constructing a decimal raises :class:`InvalidOperation`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 199); *backlink*

Unknown interpreted text role "class".

```
>>> Decimal("1e9999999999999999999")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
decimal.InvalidOperation: [<class 'decimal.InvalidOperation'>]
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 207)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.3
```

Decimals interact well with much of the rest of Python. Here is a small decimal floating point flying circus:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 212)

Unknown directive type "doctest".

```
.. doctest::
   :options: +NORMALIZE_WHITESPACE

   >>> data = list(map(Decimal, '1.34 1.87 3.45 2.35 1.00 0.03 9.25'.split()))
   >>> max(data)
   Decimal('9.25')
   >>> min(data)
   Decimal('0.03')
   >>> sorted(data)
   [Decimal('0.03'), Decimal('1.00'), Decimal('1.34'), Decimal('1.87'),
    Decimal('2.35'), Decimal('3.45'), Decimal('9.25')]
   >>> sum(data)
   Decimal('19.29')
   >>> a,b,c = data[:3]
```

```
        >>> str(a)
        '1.34'
        >>> float(a)
        1.34
        >>> round(a, 1)
        Decimal('1.3')
        >>> int(a)
        1
        >>> a * 5
        Decimal('6.70')
        >>> a * b
        Decimal('2.5058')
        >>> c % a
        Decimal('0.77')
```

And some mathematical functions are also available to Decimal:

```
>>> getcontext().prec = 28
>>> Decimal(2).sqrt()
Decimal('1.414213562373095048801688724')
>>> Decimal(1).exp()
Decimal('2.718281828459045235360287471')
>>> Decimal('10').ln()
Decimal('2.302585092994045684017991455')
>>> Decimal('10').log10()
Decimal('1')
```

The :meth:`quantize` method rounds a number to a fixed exponent. This method is useful for monetary applications that often round results to a fixed number of places:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 253); *backlink*

Unknown interpreted text role "meth".

```
>>> Decimal('7.325').quantize(Decimal('.01'), rounding=ROUND_DOWN)
Decimal('7.32')
>>> Decimal('7.325').quantize(Decimal('1.'), rounding=ROUND_UP)
Decimal('8')
```

As shown above, the :func:`getcontext` function accesses the current context and allows the settings to be changed. This approach meets the needs of most applications.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 262); *backlink*

Unknown interpreted text role "func".

For more advanced work, it may be useful to create alternate contexts using the Context() constructor. To make an alternate active, use the :func:`setcontext` function.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 266); *backlink*

Unknown interpreted text role "func".

In accordance with the standard, the :mod:`decimal` module provides two ready to use standard contexts, :const:`BasicContext` and :const:`ExtendedContext`. The former is especially useful for debugging because many of the traps are enabled:

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

Unknown interpreted text role "mod".

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

Unknown interpreted text role "const".

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 270); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 275)

Unknown directive type "doctest".

```
.. doctest:: newcontext
   :options: +NORMALIZE_WHITESPACE

   >>> myothercontext = Context(prec=60, rounding=ROUND_HALF_DOWN)
   >>> setcontext(myothercontext)
   >>> Decimal(1) / Decimal(7)
   Decimal('0.142857142857142857142857142857142857142857142857142857142857')

   >>> ExtendedContext
   Context(prec=9, rounding=ROUND_HALF_EVEN, Emin=-999999, Emax=999999,
           capitals=1, clamp=0, flags=[], traps=[])
   >>> setcontext(ExtendedContext)
   >>> Decimal(1) / Decimal(7)
   Decimal('0.142857143')
   >>> Decimal(42) / Decimal(0)
   Decimal('Infinity')

   >>> setcontext(BasicContext)
   >>> Decimal(42) / Decimal(0)
   Traceback (most recent call last):
     File "<pyshell#143>", line 1, in -toplevel-
       Decimal(42) / Decimal(0)
   DivisionByZero: x / 0
```

Contexts also have signal flags for monitoring exceptional conditions encountered during computations. The flags remain set until explicitly cleared, so it is best to clear the flags before each set of monitored computations by using the :meth:`clear_flags` method.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 299); *backlink*

Unknown interpreted text role "meth".

```
>>> setcontext(ExtendedContext)
>>> getcontext().clear_flags()
>>> Decimal(355) / Decimal(113)
Decimal('3.14159292')
>>> getcontext()
Context(prec=9, rounding=ROUND_HALF_EVEN, Emin=-999999, Emax=999999,
        capitals=1, clamp=0, flags=[Inexact, Rounded], traps=[])
```

The *flags* entry shows that the rational approximation to :const:`Pi` was rounded (digits beyond the context precision were thrown away) and that the result is inexact (some of the discarded digits were non-zero).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 312); *backlink*

Unknown interpreted text role "const".

Individual traps are set using the dictionary in the :attr:`traps` field of a context:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 316); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 319)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   >>> setcontext(ExtendedContext)
   >>> Decimal(1) / Decimal(0)
```

```
Decimal('Infinity')
>>> getcontext().traps[DivisionByZero] = 1
>>> Decimal(1) / Decimal(0)
Traceback (most recent call last):
  File "<pyshell#112>", line 1, in -toplevel-
    Decimal(1) / Decimal(0)
DivisionByZero: x / 0
```

Most programs adjust the current context only once, at the beginning of the program. And, in many applications, data is converted to :class:`Decimal` with a single cast inside a loop. With context set and decimals created, the bulk of the program manipulates the data no differently than with other Python numeric types.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 331); *backlink*

Unknown interpreted text role "class".

Decimal objects

Construct a new :class:`Decimal` object based from *value*.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 348); *backlink*

Unknown interpreted text role "class".

value can be an integer, string, tuple, :class:`float`, or another :class:`Decimal` object. If no *value* is given, returns `Decimal('0')`. If *value* is a string, it should conform to the decimal numeric string syntax after leading and trailing whitespace characters, as well as underscores throughout, are removed:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 350); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 350); *backlink*

Unknown interpreted text role "class".

```
sign           ::=  '+' | '-'
digit          ::=  '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
indicator      ::=  'e' | 'E'
digits         ::=  digit [digit]...
decimal-part   ::=  digits '.' [digits] | ['.'] digits
exponent-part  ::=  indicator [sign] digits
infinity       ::=  'Infinity' | 'Inf'
nan            ::=  'NaN' [digits] | 'sNaN' [digits]
numeric-value  ::=  decimal-part [exponent-part] | infinity
numeric-string ::=  [sign] numeric-value | [sign] nan
```

Other Unicode decimal digits are also permitted where `digit` appears above. These include decimal digits from various other alphabets (for example, Arabic-Indic and Devanāgarī digits) along with the fullwidth digits `'\uff10'` through `'\uff19'`.

If *value* is a :class:`tuple`, it should have three components, a sign (:const:`0` for positive or :const:`1` for negative), a :class:`tuple` of digits, and an integer exponent. For example, `Decimal((0, (1, 4, 1, 4), -3))` returns `Decimal('1.414')`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 371); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 371); *backlink*

Unknown interpreted text role "const".

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 371); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 371); *backlink*

Unknown interpreted text role "class".

If *value* is a :class:`float`, the binary floating point value is losslessly converted to its exact decimal equivalent. This conversion can often require 53 or more digits of precision. For example, `Decimal(float('1.1'))` converts to `Decimal('1.100000000000000088817841970012523233890533447265625')`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 376); *backlink*

Unknown interpreted text role "class".

The *context* precision does not affect how many digits are stored. That is determined exclusively by the number of digits in *value*. For example, `Decimal('3.00000')` records all five zeros even if the context precision is only three.

The purpose of the *context* argument is determining what to do if *value* is a malformed string. If the context traps :const:`InvalidOperation`, an exception is raised; otherwise, the constructor returns a new Decimal with the value of :const:`NaN`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 387); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 387); *backlink*

Unknown interpreted text role "const".

Once constructed, :class:`Decimal` objects are immutable.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 392); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 394)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   The argument to the constructor is now permitted to be a :class:`float`
   instance.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 398)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.3
   :class:`float` arguments raise an exception if the :exc:`FloatOperation`
   trap is set. By default the trap is off.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 402)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.6
   Underscores are allowed for grouping, as with integral and floating-point
   literals in code.
```

Decimal floating point objects share many properties with the other built-in numeric types such as :class:`float` and :class:`int`. All of the usual math operations and special methods apply. Likewise, decimal objects can be copied, pickled, printed, used as dictionary keys, used as set elements, compared, sorted, and coerced to another type (such as :class:`float` or :class:`int`).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 406); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 406); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 406); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 406); *backlink*

Unknown interpreted text role "class".

There are some small differences between arithmetic on Decimal objects and arithmetic on integers and floats. When the remainder operator `%` is applied to Decimal objects, the sign of the result is the sign of the *dividend* rather than the sign of the divisor:

```
>>> (-7) % 4
1
>>> Decimal(-7) % Decimal(4)
Decimal('-3')
```

The integer division operator `//` behaves analogously, returning the integer part of the true quotient (truncating towards zero) rather than its floor, so as to preserve the usual identity `x == (x // y) * y + x % y`:

```
>>> -7 // 4
-2
>>> Decimal(-7) // Decimal(4)
Decimal('-1')
```

The `%` and `//` operators implement the `remainder` and `divide-integer` operations (respectively) as described in the specification.

Decimal objects cannot generally be combined with floats or instances of :class:`fractions.Fraction` in arithmetic operations: an attempt to add a :class:`Decimal` to a :class:`float`, for example, will raise a :exc:`TypeError`. However, it is possible to use Python's comparison operators to compare a :class:`Decimal` instance `x` with another number `y`. This avoids confusing results when doing equality comparisons between numbers of different types.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 436); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 436); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 436); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 436); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 436); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 444)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.2
   Mixed-type comparisons between :class:`Decimal` instances and other
   numeric types are now fully supported.
```

In addition to the standard numeric properties, decimal floating point objects also have a number of specialized methods:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 452)

Unknown directive type "method".

```
.. method:: adjusted()

   Return the adjusted exponent after shifting out the coefficient's
   rightmost digits until only the lead digit remains:
   ``Decimal('321e+5').adjusted()`` returns seven.  Used for determining the
   position of the most significant digit with respect to the decimal point.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 459)

Unknown directive type "method".

```
.. method:: as_integer_ratio()

   Return a pair ``(n, d)`` of integers that represent the given
   :class:`Decimal` instance as a fraction, in lowest terms and
   with a positive denominator::

       >>> Decimal('-3.14').as_integer_ratio()
       (-157, 50)

   The conversion is exact.  Raise OverflowError on infinities and ValueError
   on NaNs.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 471)

Unknown directive type "versionadded".

```
.. versionadded:: 3.6
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 473)

Unknown directive type "method".

```
.. method:: as_tuple()

   Return a :term:`named tuple` representation of the number:
   ``DecimalTuple(sign, digits, exponent)``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 479)

Unknown directive type "method".

```
.. method:: canonical()
```

```
   Return the canonical encoding of the argument.  Currently, the encoding of
   a :class:`Decimal` instance is always canonical, so this operation returns
   its argument unchanged.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 485)

Unknown directive type "method".

```
.. method:: compare(other, context=None)

   Compare the values of two Decimal instances.  :meth:`compare` returns a
   Decimal instance, and if either operand is a NaN then the result is a
   NaN::

      a or b is a NaN  ==> Decimal('NaN')
      a < b            ==> Decimal('-1')
      a == b           ==> Decimal('0')
      a > b            ==> Decimal('1')
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 496)

Unknown directive type "method".

```
.. method:: compare_signal(other, context=None)

   This operation is identical to the :meth:`compare` method, except that all
   NaNs signal.  That is, if neither operand is a signaling NaN then any
   quiet NaN operand is treated as though it were a signaling NaN.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 502)

Unknown directive type "method".

```
.. method:: compare_total(other, context=None)

   Compare two operands using their abstract representation rather than their
   numerical value.  Similar to the :meth:`compare` method, but the result
   gives a total ordering on :class:`Decimal` instances.  Two
   :class:`Decimal` instances with the same numeric value but different
   representations compare unequal in this ordering:

      >>> Decimal('12.0').compare_total(Decimal('12'))
      Decimal('-1')

   Quiet and signaling NaNs are also included in the total ordering.  The
   result of this function is ``Decimal('0')`` if both operands have the same
   representation, ``Decimal('-1')`` if the first operand is lower in the
   total order than the second, and ``Decimal('1')`` if the first operand is
   higher in the total order than the second operand.  See the specification
   for details of the total order.

   This operation is unaffected by context and is quiet: no flags are changed
   and no rounding is performed.  As an exception, the C version may raise
   InvalidOperation if the second operand cannot be converted exactly.
```

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

Unknown directive type "method".

```
.. method:: compare_total_mag(other, context=None)

   Compare two operands using their abstract representation rather than their
   value as in :meth:`compare_total`, but ignoring the sign of each operand.
   ``x.compare_total_mag(y)`` is equivalent to
   ``x.copy_abs().compare_total(y.copy_abs())``.

   This operation is unaffected by context and is quiet: no flags are changed
   and no rounding is performed.  As an exception, the C version may raise
```

```
    InvalidOperation if the second operand cannot be converted exactly.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 535)

Unknown directive type "method".

```
.. method:: conjugate()

   Just returns self, this method is only to comply with the Decimal
   Specification.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 540)

Unknown directive type "method".

```
.. method:: copy_abs()

   Return the absolute value of the argument.  This operation is unaffected
   by the context and is quiet: no flags are changed and no rounding is
   performed.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 546)

Unknown directive type "method".

```
.. method:: copy_negate()

   Return the negation of the argument.  This operation is unaffected by the
   context and is quiet: no flags are changed and no rounding is performed.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 551)

Unknown directive type "method".

```
.. method:: copy_sign(other, context=None)

   Return a copy of the first operand with the sign set to be the same as the
   sign of the second operand.  For example:

      >>> Decimal('2.3').copy_sign(Decimal('-1.5'))
      Decimal('-2.3')

   This operation is unaffected by context and is quiet: no flags are changed
   and no rounding is performed.  As an exception, the C version may raise
   InvalidOperation if the second operand cannot be converted exactly.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 563)

Unknown directive type "method".

```
.. method:: exp(context=None)

   Return the value of the (natural) exponential function ``e**x`` at the
   given number.  The result is correctly rounded using the
   :const:`ROUND_HALF_EVEN` rounding mode.

   >>> Decimal(1).exp()
   Decimal('2.718281828459045235360287471')
   >>> Decimal(321).exp()
   Decimal('2.561702493119680037517373933E+139')
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 574)

Unknown directive type "method".

```
.. method:: from_float(f)

   Classmethod that converts a float to a decimal number, exactly.

   Note `Decimal.from_float(0.1)` is not the same as `Decimal('0.1')`.
   Since 0.1 is not exactly representable in binary floating point, the
   value is stored as the nearest representable value which is
   `0x1.999999999999ap-4`.  That equivalent value in decimal is
   `0.1000000000000000055511151231257827021181583404541015625`.

   .. note:: From Python 3.2 onwards, a :class:`Decimal` instance
      can also be constructed directly from a :class:`float`.

   .. doctest::

       >>> Decimal.from_float(0.1)
       Decimal('0.1000000000000000055511151231257827021181583404541015625')
       >>> Decimal.from_float(float('nan'))
       Decimal('NaN')
       >>> Decimal.from_float(float('inf'))
       Decimal('Infinity')
       >>> Decimal.from_float(float('-inf'))
       Decimal('-Infinity')

   .. versionadded:: 3.1
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 600)

Unknown directive type "method".

```
.. method:: fma(other, third, context=None)

   Fused multiply-add.  Return self*other+third with no rounding of the
   intermediate product self*other.

   >>> Decimal(2).fma(3, 5)
   Decimal('11')
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 608)

Unknown directive type "method".

```
.. method:: is_canonical()

   Return :const:`True` if the argument is canonical and :const:`False`
   otherwise.  Currently, a :class:`Decimal` instance is always canonical, so
   this operation always returns :const:`True`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 614)

Unknown directive type "method".

```
.. method:: is_finite()

   Return :const:`True` if the argument is a finite number, and
   :const:`False` if the argument is an infinity or a NaN.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst`, line 619)

Unknown directive type "method".

```
.. method:: is_infinite()

   Return :const:`True` if the argument is either positive or negative
   infinity and :const:`False` otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 624)

Unknown directive type "method".

```
.. method:: is_nan()

   Return :const:`True` if the argument is a (quiet or signaling) NaN and
   :const:`False` otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 629)

Unknown directive type "method".

```
.. method:: is_normal(context=None)

   Return :const:`True` if the argument is a *normal* finite number.  Return
   :const:`False` if the argument is zero, subnormal, infinite or a NaN.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 634)

Unknown directive type "method".

```
.. method:: is_qnan()

   Return :const:`True` if the argument is a quiet NaN, and
   :const:`False` otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 639)

Unknown directive type "method".

```
.. method:: is_signed()

   Return :const:`True` if the argument has a negative sign and
   :const:`False` otherwise.  Note that zeros and NaNs can both carry signs.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 644)

Unknown directive type "method".

```
.. method:: is_snan()

   Return :const:`True` if the argument is a signaling NaN and :const:`False`
   otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 649)

Unknown directive type "method".

```
.. method:: is_subnormal(context=None)

   Return :const:`True` if the argument is subnormal, and :const:`False`
   otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 654)

Unknown directive type "method".

```
.. method:: is_zero()

   Return :const:`True` if the argument is a (positive or negative) zero and
   :const:`False` otherwise.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 659)

Unknown directive type "method".

```
.. method:: ln(context=None)

   Return the natural (base e) logarithm of the operand.  The result is
   correctly rounded using the :const:`ROUND_HALF_EVEN` rounding mode.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 664)

Unknown directive type "method".

```
.. method:: log10(context=None)

   Return the base ten logarithm of the operand.  The result is correctly
   rounded using the :const:`ROUND_HALF_EVEN` rounding mode.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 669)

Unknown directive type "method".

```
.. method:: logb(context=None)

   For a nonzero number, return the adjusted exponent of its operand as a
   :class:`Decimal` instance.  If the operand is a zero then
   ``Decimal('-Infinity')`` is returned and the :const:`DivisionByZero` flag
   is raised.  If the operand is an infinity then ``Decimal('Infinity')`` is
   returned.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 677)

Unknown directive type "method".

```
.. method:: logical_and(other, context=None)

   :meth:`logical_and` is a logical operation which takes two *logical
   operands* (see :ref:`logical_operands_label`).  The result is the
   digit-wise ``and`` of the two operands.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 683)

Unknown directive type "method".

```
.. method:: logical_invert(context=None)

   :meth:`logical_invert` is a logical operation.  The
   result is the digit-wise inversion of the operand.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 688)

Unknown directive type "method".

```
.. method:: logical_or(other, context=None)

   :meth:`logical_or` is a logical operation which takes two *logical
   operands* (see :ref:`logical_operands_label`).  The result is the
   digit-wise ``or`` of the two operands.
```

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-`

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 694)

Unknown directive type "method".

```
.. method:: logical_xor(other, context=None)

   :meth:`logical_xor` is a logical operation which takes two *logical
   operands* (see :ref:`logical_operands_label`).  The result is the
   digit-wise exclusive or of the two operands.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 700)

Unknown directive type "method".

```
.. method:: max(other, context=None)

   Like ``max(self, other)`` except that the context rounding rule is applied
   before returning and that :const:`NaN` values are either signaled or
   ignored (depending on the context and whether they are signaling or
   quiet).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 707)

Unknown directive type "method".

```
.. method:: max_mag(other, context=None)

   Similar to the :meth:`.max` method, but the comparison is done using the
   absolute values of the operands.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 712)

Unknown directive type "method".

```
.. method:: min(other, context=None)

   Like ``min(self, other)`` except that the context rounding rule is applied
   before returning and that :const:`NaN` values are either signaled or
   ignored (depending on the context and whether they are signaling or
   quiet).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 719)

Unknown directive type "method".

```
.. method:: min_mag(other, context=None)

   Similar to the :meth:`.min` method, but the comparison is done using the
   absolute values of the operands.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 724)

Unknown directive type "method".

```
.. method:: next_minus(context=None)

   Return the largest number representable in the given context (or in the
   current thread's context if no context is given) that is smaller than the
   given operand.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 730)

Unknown directive type "method".

```
.. method:: next_plus(context=None)

   Return the smallest number representable in the given context (or in the
   current thread's context if no context is given) that is larger than the
   given operand.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 736)

Unknown directive type "method".

```
.. method:: next_toward(other, context=None)

   If the two operands are unequal, return the number closest to the first
   operand in the direction of the second operand.  If both operands are
   numerically equal, return a copy of the first operand with the sign set to
   be the same as the sign of the second operand.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 743)

Unknown directive type "method".

```
.. method:: normalize(context=None)

   Normalize the number by stripping the rightmost trailing zeros and
   converting any result equal to :const:`Decimal('0')` to
   :const:`Decimal('0e0')`. Used for producing canonical values for attributes
   of an equivalence class. For example, ``Decimal('32.100')`` and
   ``Decimal('0.321000e+2')`` both normalize to the equivalent value
   ``Decimal('32.1')``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 752)

Unknown directive type "method".

```
.. method:: number_class(context=None)

   Return a string describing the *class* of the operand.  The returned value
   is one of the following ten strings.

   * ``"-Infinity"``, indicating that the operand is negative infinity.
   * ``"-Normal"``, indicating that the operand is a negative normal number.
   * ``"-Subnormal"``, indicating that the operand is negative and subnormal.
   * ``"-Zero"``, indicating that the operand is a negative zero.
   * ``"+Zero"``, indicating that the operand is a positive zero.
   * ``"+Subnormal"``, indicating that the operand is positive and subnormal.
   * ``"+Normal"``, indicating that the operand is a positive normal number.
   * ``"+Infinity"``, indicating that the operand is positive infinity.
   * ``"NaN"``, indicating that the operand is a quiet NaN (Not a Number).
   * ``"sNaN"``, indicating that the operand is a signaling NaN.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 768)

Unknown directive type "method".

```
.. method:: quantize(exp, rounding=None, context=None)

   Return a value equal to the first operand after rounding and having the
   exponent of the second operand.

   >>> Decimal('1.41421356').quantize(Decimal('1.000'))
   Decimal('1.414')

   Unlike other operations, if the length of the coefficient after the
   quantize operation would be greater than precision, then an
   :const:`InvalidOperation` is signaled. This guarantees that, unless there
   is an error condition, the quantized exponent is always equal to that of
   the right-hand operand.

   Also unlike other operations, quantize never signals Underflow, even if
   the result is subnormal and inexact.

   If the exponent of the second operand is larger than that of the first
   then rounding may be necessary.  In this case, the rounding mode is
   determined by the ``rounding`` argument if given, else by the given
   ``context`` argument; if neither argument is given the rounding mode of
   the current thread's context is used.

   An error is returned whenever the resulting exponent is greater than
   :attr:`Emax` or less than :attr:`Etiny`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 794)

Unknown directive type "method".

```
.. method:: radix()

   Return ``Decimal(10)``, the radix (base) in which the :class:`Decimal`
   class does all its arithmetic.  Included for compatibility with the
   specification.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 800)

Unknown directive type "method".

```
.. method:: remainder_near(other, context=None)

   Return the remainder from dividing *self* by *other*.  This differs from
   ``self % other`` in that the sign of the remainder is chosen so as to
   minimize its absolute value.  More precisely, the return value is
   ``self - n * other`` where ``n`` is the integer nearest to the exact
   value of ``self / other``, and if two integers are equally near then the
   even one is chosen.

   If the result is zero then its sign will be the sign of *self*.

   >>> Decimal(18).remainder_near(Decimal(10))
   Decimal('-2')
   >>> Decimal(25).remainder_near(Decimal(10))
   Decimal('5')
   >>> Decimal(35).remainder_near(Decimal(10))
   Decimal('-5')
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 818)

Unknown directive type "method".

```
.. method:: rotate(other, context=None)

   Return the result of rotating the digits of the first operand by an amount
   specified by the second operand.  The second operand must be an integer in
   the range -precision through precision.  The absolute value of the second
   operand gives the number of places to rotate.  If the second operand is
   positive then rotation is to the left; otherwise rotation is to the right.
   The coefficient of the first operand is padded on the left with zeros to
   length precision if necessary.  The sign and exponent of the first operand
   are unchanged.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 829)

Unknown directive type "method".

```
.. method:: same_quantum(other, context=None)

   Test whether self and other have the same exponent or whether both are
   :const:`NaN`.

   This operation is unaffected by context and is quiet: no flags are changed
   and no rounding is performed.  As an exception, the C version may raise
```

```
      InvalidOperation if the second operand cannot be converted exactly.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 838)

Unknown directive type "method".

```
.. method:: scaleb(other, context=None)

   Return the first operand with exponent adjusted by the second.
   Equivalently, return the first operand multiplied by ``10**other``.  The
   second operand must be an integer.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 844)

Unknown directive type "method".

```
.. method:: shift(other, context=None)

   Return the result of shifting the digits of the first operand by an amount
   specified by the second operand.  The second operand must be an integer in
   the range -precision through precision.  The absolute value of the second
   operand gives the number of places to shift.  If the second operand is
   positive then the shift is to the left; otherwise the shift is to the
   right.  Digits shifted into the coefficient are zeros.  The sign and
   exponent of the first operand are unchanged.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 854)

Unknown directive type "method".

```
.. method:: sqrt(context=None)

   Return the square root of the argument to full precision.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 859)

Unknown directive type "method".

```
.. method:: to_eng_string(context=None)

   Convert to a string, using engineering notation if an exponent is needed.

   Engineering notation has an exponent which is a multiple of 3.  This
   can leave up to 3 digits to the left of the decimal place and may
   require the addition of either one or two trailing zeros.

   For example, this converts ``Decimal('123E+1')`` to ``Decimal('1.23E+3')``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 869)

Unknown directive type "method".

```
.. method:: to_integral(rounding=None, context=None)

   Identical to the :meth:`to_integral_value` method.  The ``to_integral``
   name has been kept for compatibility with older versions.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 874)

Unknown directive type "method".

```
.. method:: to_integral_exact(rounding=None, context=None)
```

```
   Round to the nearest integer, signaling :const:`Inexact` or
   :const:`Rounded` as appropriate if rounding occurs.  The rounding mode is
   determined by the ``rounding`` parameter if given, else by the given
   ``context``.  If neither parameter is given then the rounding mode of the
   current context is used.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 882)

Unknown directive type "method".

```
.. method:: to_integral_value(rounding=None, context=None)

   Round to the nearest integer without signaling :const:`Inexact` or
   :const:`Rounded`.  If given, applies *rounding*; otherwise, uses the
   rounding method in either the supplied *context* or the current context.
```

Logical operands

The :meth:`logical_and`, :meth:`logical_invert`, :meth:`logical_or`, and :meth:`logical_xor` methods expect their arguments to be *logical operands*. A *logical operand* is a :class:`Decimal` instance whose exponent and sign are both zero, and whose digits are all either :const:`0` or :const:`1`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 894); *backlink*

Unknown interpreted text role "const".

Context objects

Contexts are environments for arithmetic operations. They govern precision, set rules for rounding, determine which signals are treated as exceptions, and limit the range for exponents.

Each thread has its own current context which is accessed or changed using the :func:`getcontext` and :func:`setcontext` functions:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 912); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 912); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 916)

Unknown directive type "function".

```
.. function:: getcontext()

   Return the current context for the active thread.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 921)

Unknown directive type "function".

```
.. function:: setcontext(c)

   Set the current context for the active thread to *c*.
```

You can also use the :keyword:`with` statement and the :func:`localcontext` function to temporarily change the active context.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 925); *backlink*

Unknown interpreted text role "keyword".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 925); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 928)

Unknown directive type "function".

```
.. function:: localcontext(ctx=None)

   Return a context manager that will set the current context for the active thread
   to a copy of *ctx* on entry to the with-statement and restore the previous context
   when exiting the with-statement. If no context is specified, a copy of the
   current context is used.

   For example, the following code sets the current decimal precision to 42 places,
   performs a calculation, and then automatically restores the previous context::

      from decimal import localcontext

      with localcontext() as ctx:
          ctx.prec = 42   # Perform a high precision calculation
          s = calculate_something()
      s = +s  # Round the final result back to the default precision
```

New contexts can also be created using the :class:`Context` constructor described below. In addition, the module provides three pre-made contexts:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 945); *backlink*

Unknown interpreted text role "class".

This is a standard context defined by the General Decimal Arithmetic Specification. Precision is set to nine. Rounding is set to :const:`ROUND_HALF_UP`. All flags are cleared. All traps are enabled (treated as exceptions) except :const:`Inexact`, :const:`Rounded`, and :const:`Subnormal`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 951**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 951**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 951**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 951**); ***backlink***

Unknown interpreted text role "const".

Because many of the traps are enabled, this context is useful for debugging.

This is a standard context defined by the General Decimal Arithmetic Specification. Precision is set to nine. Rounding is set to :const:`ROUND_HALF_EVEN`. All flags are cleared. No traps are enabled (so that exceptions are not raised during computations).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 962**); ***backlink***

Unknown interpreted text role "const".

Because the traps are disabled, this context is useful for applications that prefer to have result value of :const:`NaN` or :const:`Infinity` instead of raising exceptions. This allows an application to complete a run in the presence of conditions that would otherwise halt the program.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 967**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 967**); ***backlink***

Unknown interpreted text role "const".

This context is used by the :class:`Context` constructor as a prototype for new contexts. Changing a field (such a precision) has the effect of changing the default for new contexts created by the :class:`Context` constructor.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 975**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 975**); ***backlink***

Unknown interpreted text role "class".

This context is most useful in multi-threaded environments. Changing one of the fields before threads are started has the effect of setting system-wide defaults. Changing the fields after threads have started is not recommended as it would require thread synchronization to prevent race conditions.

In single threaded environments, it is preferable to not use this context at all. Instead, simply create contexts explicitly as described below.

The default values are :attr:`prec`=:const:`28`, :attr:`rounding`=:const:`ROUND_HALF_EVEN`, and enabled traps for :class:`Overflow`, :class:`InvalidOperation`, and :class:`DivisionByZero`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 987); *backlink*

Unknown interpreted text role "class".

In addition to the three supplied contexts, new contexts can be created with the :class:`Context` constructor.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 992); *backlink*

Unknown interpreted text role "class".

Creates a new context. If a field is not specified or is :const:`None`, the default values are copied from the :const:`DefaultContext`. If the *flags* field is not specified or is :const:`None`, all flags are cleared.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 998); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 998); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-`

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 998); *backlink*

Unknown interpreted text role "const".

prec is an integer in the range [:const:`1`, :const:`MAX_PREC`] that sets the precision for arithmetic operations in the context.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1002); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1002); *backlink*

Unknown interpreted text role "const".

The *rounding* option is one of the constants listed in the section Rounding Modes.

The *traps* and *flags* fields list any signals to be set. Generally, new contexts should only set traps and leave the flags clear.

The *Emin* and *Emax* fields are integers specifying the outer limits allowable for exponents. *Emin* must be in the range [:const:`MIN_EMIN`, :const:`0`], *Emax* in the range [:const:`0`, :const:`MAX_EMAX`].

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1011); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1011); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1011); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1011); *backlink*

Unknown interpreted text role "const".

The *capitals* field is either :const:`0` or :const:`1` (the default). If set to :const:`1`, exponents are printed with a capital :const:`E`; otherwise, a lowercase :const:`e` is used: :const:`Decimal('6.02e+23')`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1015); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1015); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1015); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1015); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1015); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1015); *backlink*

Unknown interpreted text role "const".

The *clamp* field is either :const:`0` (the default) or :const:`1`. If set to :const:`1`, the exponent `e` of a :class:`Decimal` instance representable in this context is strictly limited to the range `Emin - prec + 1 <= e <= Emax - prec + 1`. If *clamp* is :const:`0` then a weaker condition holds: the adjusted exponent of the :class:`Decimal` instance is at most `Emax`. When *clamp* is :const:`1`, a large normal number will, where possible, have its exponent reduced and a corresponding number of zeros added to its coefficient, in order to fit the exponent constraints; this preserves the value of the number but loses information about significant trailing zeros. For example:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1019); *backlink*

Unknown interpreted text role "const".

```
>>> Context(prec=6, Emax=999, clamp=1).create_decimal('1.23e999')
Decimal('1.23000E+999')
```

A *clamp* value of :const:`1` allows compatibility with the fixed-width decimal interchange formats specified in IEEE 754.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1034); *backlink*

Unknown interpreted text role "const".

The :class:`Context` class defines several general purpose methods as well as a large number of methods for doing arithmetic directly in a given context. In addition, for each of the :class:`Decimal` methods described above (with the exception of the :meth:`adjusted` and :meth:`as_tuple` methods) there is a corresponding :class:`Context` method. For example, for a :class:`Context` instance `C` and

:class:`Decimal` instance `x`, `C.exp(x)` is equivalent to `x.exp(context=C)`. Each :class:`Context` method accepts a Python integer (an instance of :class:`int`) anywhere that a Decimal instance is accepted.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1037); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1048)

Unknown directive type "method".

```
.. method:: clear_flags()

   Resets all of the flags to :const:`0`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1052)

Unknown directive type "method".

```
.. method:: clear_traps()

   Resets all of the traps to :const:`0`.
   .. versionadded:: 3.3
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1058)

Unknown directive type "method".

```
.. method:: copy()

   Return a duplicate of the context.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1062)

Unknown directive type "method".

```
.. method:: copy_decimal(num)

   Return a copy of the Decimal instance num.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1066)

Unknown directive type "method".

```
.. method:: create_decimal(num)

   Creates a new Decimal instance from *num* but using *self* as
   context. Unlike the :class:`Decimal` constructor, the context precision,
   rounding method, flags, and traps are applied to the conversion.

   This is useful because constants are often given to a greater precision
   than is needed by the application.  Another benefit is that rounding
   immediately eliminates unintended effects from digits beyond the current
   precision. In the following example, using unrounded inputs means that
   adding zero to a sum can change the result:

   .. doctest:: newcontext

      >>> getcontext().prec = 3
      >>> Decimal('3.4445') + Decimal('1.0023')
      Decimal('4.45')
      >>> Decimal('3.4445') + Decimal(0) + Decimal('1.0023')
      Decimal('4.44')

   This method implements the to-number operation of the IBM specification.
   If the argument is a string, no leading or trailing whitespace or
   underscores are permitted.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1090)

Unknown directive type "method".

```
.. method:: create_decimal_from_float(f)

   Creates a new Decimal instance from a float *f* but rounding using *self*
   as the context.  Unlike the :meth:`Decimal.from_float` class method,
   the context precision, rounding method, flags, and traps are applied to
   the conversion.

   .. doctest::

      >>> context = Context(prec=5, rounding=ROUND_DOWN)
      >>> context.create_decimal_from_float(math.pi)
      Decimal('3.1415')
      >>> context = Context(prec=5, traps=[Inexact])
      >>> context.create_decimal_from_float(math.pi)
      Traceback (most recent call last):
          ...
      decimal.Inexact: None

   .. versionadded:: 3.1
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1110)

Unknown directive type "method".

```
.. method:: Etiny()

   Returns a value equal to ``Emin - prec + 1`` which is the minimum exponent
   value for subnormal results.  When underflow occurs, the exponent is set
   to :const:`Etiny`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1116)

Unknown directive type "method".

```
.. method:: Etop()

   Returns a value equal to ``Emax - prec + 1``.
```

The usual approach to working with decimals is to create :class:`Decimal` instances and then apply arithmetic operations which take place within the current context for the active thread. An alternative approach is to use context methods for calculating within a specific context. The methods are similar to those for the :class:`Decimal` class and are only briefly recounted here.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1120); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1120); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1128)

Unknown directive type "method".

```
.. method:: abs(x)

   Returns the absolute value of *x*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1133)

Unknown directive type "method".

```
.. method:: add(x, y)

   Return the sum of *x* and *y*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1138)

Unknown directive type "method".

```
.. method:: canonical(x)

   Returns the same Decimal object *x*.
```

**System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1143)

Unknown directive type "method".

```
.. method:: compare(x, y)

   Compares *x* and *y* numerically.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1148)

Unknown directive type "method".

```
.. method:: compare_signal(x, y)

   Compares the values of the two operands numerically.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1153)

Unknown directive type "method".

```
.. method:: compare_total(x, y)

   Compares two operands using their abstract representation.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1158)

Unknown directive type "method".

```
.. method:: compare_total_mag(x, y)

   Compares two operands using their abstract representation, ignoring sign.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1163)

Unknown directive type "method".

```
.. method:: copy_abs(x)

   Returns a copy of *x* with the sign set to 0.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1168)

Unknown directive type "method".

```
.. method:: copy_negate(x)

   Returns a copy of *x* with the sign inverted.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1173)

Unknown directive type "method".

```
.. method:: copy_sign(x, y)

   Copies the sign from *y* to *x*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1178)

Unknown directive type "method".

```
.. method:: divide(x, y)

   Return *x* divided by *y*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1183)

Unknown directive type "method".

```
.. method:: divide_int(x, y)

   Return *x* divided by *y*, truncated to an integer.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1188)

Unknown directive type "method".

```
.. method:: divmod(x, y)

   Divides two numbers and returns the integer part of the result.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1193)

Unknown directive type "method".

```
.. method:: exp(x)

   Returns `e ** x`.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1198)

Unknown directive type "method".

```
.. method:: fma(x, y, z)

   Returns *x* multiplied by *y*, plus *z*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1203)

Unknown directive type "method".

```
.. method:: is_canonical(x)

   Returns ``True`` if *x* is canonical; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1208)

Unknown directive type "method".

```
.. method:: is_finite(x)

   Returns ``True`` if *x* is finite; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1213)

Unknown directive type "method".

```
.. method:: is_infinite(x)

   Returns ``True`` if *x* is infinite; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1218)

Unknown directive type "method".

```
.. method:: is_nan(x)

   Returns ``True`` if *x* is a qNaN or sNaN; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1223)

Unknown directive type "method".

```
.. method:: is_normal(x)

   Returns ``True`` if *x* is a normal number; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1228)

Unknown directive type "method".

```
.. method:: is_qnan(x)

   Returns ``True`` if *x* is a quiet NaN; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1233)

Unknown directive type "method".

```
.. method:: is_signed(x)

   Returns ``True`` if *x* is negative; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1238)

Unknown directive type "method".

```
.. method:: is_snan(x)

   Returns ``True`` if *x* is a signaling NaN; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1243)

Unknown directive type "method".

```
.. method:: is_subnormal(x)
```

```
    Returns ``True`` if *x* is subnormal; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1248)

Unknown directive type "method".

```
.. method:: is_zero(x)

   Returns ``True`` if *x* is a zero; otherwise returns ``False``.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1253)

Unknown directive type "method".

```
.. method:: ln(x)

   Returns the natural (base e) logarithm of *x*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1258)

Unknown directive type "method".

```
.. method:: log10(x)

   Returns the base 10 logarithm of *x*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1263)

Unknown directive type "method".

```
.. method:: logb(x)

    Returns the exponent of the magnitude of the operand's MSD.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1268)

Unknown directive type "method".

```
.. method:: logical_and(x, y)

   Applies the logical operation *and* between each operand's digits.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1273)

Unknown directive type "method".

```
.. method:: logical_invert(x)

   Invert all the digits in *x*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1278)

Unknown directive type "method".

```
.. method:: logical_or(x, y)

   Applies the logical operation *or* between each operand's digits.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1283)

Unknown directive type "method".

```
.. method:: logical_xor(x, y)

   Applies the logical operation *xor* between each operand's digits.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1288)

Unknown directive type "method".

```
.. method:: max(x, y)

   Compares two values numerically and returns the maximum.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1293)

Unknown directive type "method".

```
.. method:: max_mag(x, y)

   Compares the values numerically with their sign ignored.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1298)

Unknown directive type "method".

```
.. method:: min(x, y)

   Compares two values numerically and returns the minimum.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1303)

Unknown directive type "method".

```
.. method:: min_mag(x, y)

   Compares the values numerically with their sign ignored.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1308)

Unknown directive type "method".

```
.. method:: minus(x)

   Minus corresponds to the unary prefix minus operator in Python.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1313)

Unknown directive type "method".

```
.. method:: multiply(x, y)

   Return the product of *x* and *y*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1318)

Unknown directive type "method".

```
.. method:: next_minus(x)

   Returns the largest representable number smaller than *x*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1323)

Unknown directive type "method".

```
.. method:: next_plus(x)

   Returns the smallest representable number larger than *x*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1328)

Unknown directive type "method".

```
.. method:: next_toward(x, y)

   Returns the number closest to *x*, in direction towards *y*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1333)

Unknown directive type "method".

```
.. method:: normalize(x)

   Reduces *x* to its simplest form.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1338)

Unknown directive type "method".

```
.. method:: number_class(x)

   Returns an indication of the class of *x*.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1343)

Unknown directive type "method".

```
.. method:: plus(x)

   Plus corresponds to the unary prefix plus operator in Python.  This
   operation applies the context precision and rounding, so it is *not* an
   identity operation.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1350)

Unknown directive type "method".

```
.. method:: power(x, y, modulo=None)

   Return ``x`` to the power of ``y``, reduced modulo ``modulo`` if given.

   With two arguments, compute ``x**y``.  If ``x`` is negative then ``y``
   must be integral.  The result will be inexact unless ``y`` is integral and
   the result is finite and can be expressed exactly in 'precision' digits.
   The rounding mode of the context is used. Results are always correctly-rounded
   in the Python version.

   ``Decimal(0) ** Decimal(0)`` results in ``InvalidOperation``, and if ``InvalidOperation``
   is not trapped, then results in ``Decimal('NaN')``.

   .. versionchanged:: 3.3
      The C module computes :meth:`power` in terms of the correctly-rounded
      :meth:`exp` and :meth:`ln` functions. The result is well-defined but
      only "almost always correctly-rounded".

   With three arguments, compute ``(x**y) % modulo``.  For the three argument
   form, the following restrictions on the arguments hold:

      - all three arguments must be integral
      - ``y`` must be nonnegative
      - at least one of ``x`` or ``y`` must be nonzero
      - ``modulo`` must be nonzero and have at most 'precision' digits

   The value resulting from ``Context.power(x, y, modulo)`` is
   equal to the value that would be obtained by computing ``(x**y)
   % modulo`` with unbounded precision, but is computed more
   efficiently.  The exponent of the result is zero, regardless of
   the exponents of ``x``, ``y`` and ``modulo``.  The result is
   always exact.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1384)

Unknown directive type "method".

```
.. method:: quantize(x, y)

   Returns a value equal to *x* (rounded), having the exponent of *y*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1389)

Unknown directive type "method".

```
.. method:: radix()

   Just returns 10, as this is Decimal, :)
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1394)

Unknown directive type "method".

```
.. method:: remainder(x, y)

   Returns the remainder from integer division.

   The sign of the result, if non-zero, is the same as that of the original
   dividend.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1402)

Unknown directive type "method".

```
.. method:: remainder_near(x, y)

   Returns ``x - y * n``, where *n* is the integer nearest the exact value
   of ``x / y`` (if the result is 0 then its sign will be the sign of *x*).
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1408)

Unknown directive type "method".

```
.. method:: rotate(x, y)

   Returns a rotated copy of *x*, *y* times.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1413)

Unknown directive type "method".

```
.. method:: same_quantum(x, y)

   Returns ``True`` if the two operands have the same exponent.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1418)

Unknown directive type "method".

```
.. method:: scaleb (x, y)

   Returns the first operand after adding the second value its exp.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1423)

Unknown directive type "method".

```
.. method:: shift(x, y)

   Returns a shifted copy of *x*, *y* times.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1428)

Unknown directive type "method".

```
.. method:: sqrt(x)

   Square root of a non-negative number to context precision.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1433)

Unknown directive type "method".

```
.. method:: subtract(x, y)

   Return the difference between *x* and *y*.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1438)

Unknown directive type "method".

```
.. method:: to_eng_string(x)

   Convert to a string, using engineering notation if an exponent is needed.

   Engineering notation has an exponent which is a multiple of 3.  This
   can leave up to 3 digits to the left of the decimal place and may
   require the addition of either one or two trailing zeros.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1447)

Unknown directive type "method".

```
.. method:: to_integral_exact(x)

   Rounds to an integer.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1452)

Unknown directive type "method".

```
.. method:: to_sci_string(x)

   Converts a number to a string using scientific notation.
```

Constants

The constants in this section are only relevant for the C module. They are also included in the pure Python version for compatibility.

	32-bit	64-bit
System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1470) Unknown directive type "data". `.. data:: MAX_PREC`	:const:\`425000000\` **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1470); *backlink*** Unknown interpreted text role "const".	:const:\`999999999999999999\` **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1470); *backlink*** Unknown interpreted text role "const".

	32-bit	64-bit
System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1472) Unknown directive type "data". `.. data:: MAX_EMAX`	:const:`425000000` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1472); *backlink* Unknown interpreted text role "const".	:const:`999999999999999999` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1472); *backlink* Unknown interpreted text role "const".
System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1474) Unknown directive type "data". `.. data:: MIN_EMIN`	:const:`-425000000` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1474); *backlink* Unknown interpreted text role "const".	:const:`-999999999999999999` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1474); *backlink* Unknown interpreted text role "const".
System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1476) Unknown directive type "data". `.. data:: MIN_ETINY`	:const:`-849999999` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1476); *backlink* Unknown interpreted text role "const".	:const:`-1999999999999999997` **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1476); *backlink* Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1479)

Unknown directive type "data".

```
.. data:: HAVE_THREADS

   The value is ``True``.  Deprecated, because Python now always has threads.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1483)

Unknown directive type "deprecated".

```
.. deprecated:: 3.9
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1485)

Unknown directive type "data".

```
.. data:: HAVE_CONTEXTVAR

   The default value is ``True``. If Python is :option:`configured using
   the --without-decimal-contextvar option <--without-decimal-contextvar>`,
   the C version uses a thread-local rather than a coroutine-local context and the value
   is ``False``.  This is slightly faster in some nested context scenarios.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1492)

Unknown directive type "versionadded".

```
.. versionadded:: 3.9 backported to 3.7 and 3.8.
```

Rounding modes

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1498)

Unknown directive type "data".

```
.. data:: ROUND_CEILING

   Round towards :const:`Infinity`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1502)

Unknown directive type "data".

```
.. data:: ROUND_DOWN

   Round towards zero.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1506)

Unknown directive type "data".

```
.. data:: ROUND_FLOOR

   Round towards :const:`-Infinity`.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1510)

Unknown directive type "data".

```
.. data:: ROUND_HALF_DOWN

   Round to nearest with ties going towards zero.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main] [Doc] [library]decimal.rst, line 1514)

Unknown directive type "data".

```
.. data:: ROUND_HALF_EVEN

   Round to nearest with ties going to nearest even integer.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1518)

Unknown directive type "data".

```
.. data:: ROUND_HALF_UP

   Round to nearest with ties going away from zero.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1522)

Unknown directive type "data".

```
.. data:: ROUND_UP

   Round away from zero.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1526)

Unknown directive type "data".

```
.. data:: ROUND_05UP

   Round away from zero if last digit after rounding towards zero would have
   been 0 or 5; otherwise round towards zero.
```

Signals

Signals represent conditions that arise during computation. Each corresponds to one context flag and one context trap enabler.

The context flag is set whenever the condition is encountered. After the computation, flags may be checked for informational purposes (for instance, to determine whether a computation was exact). After checking the flags, be sure to clear all flags before starting the next computation.

If the context's trap enabler is set for the signal, then the condition causes a Python exception to be raised. For example, if the :class:`DivisionByZero` trap is set, then a :exc:`DivisionByZero` exception is raised upon encountering the condition.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1545); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1545); *backlink*

Unknown interpreted text role "exc".

Altered an exponent to fit representation constraints.

Typically, clamping occurs when an exponent falls outside the context's :attr:`Emin` and :attr:`Emax` limits. If possible, the exponent is reduced to fit by adding zeros to the coefficient.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1555); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1555); *backlink*

Unknown interpreted text role "attr".

Base class for other signals and a subclass of :exc:`ArithmeticError`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1562**); ***backlink***

Unknown interpreted text role "exc".

Signals the division of a non-infinite number by zero.

Can occur with division, modulo division, or when raising a number to a negative power. If this signal is not trapped, returns :const:`Infinity` or :const:`-Infinity` with the sign determined by the inputs to the calculation.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1569**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1569**); ***backlink***

Unknown interpreted text role "const".

Indicates that rounding occurred and the result is not exact.

Signals when non-zero digits were discarded during rounding. The rounded result is returned. The signal flag or trap is used to detect when results are inexact.

An invalid operation was performed.

Indicates that an operation was requested that does not make sense. If not trapped, returns :const:`NaN`. Possible causes include:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1587**); ***backlink***

Unknown interpreted text role "const".

```
Infinity - Infinity
0 * Infinity
Infinity / Infinity
x % 0
Infinity % x
sqrt(-x) and x > 0
0 ** 0
x ** (non-integer)
x ** Infinity
```

Numerical overflow.

Indicates the exponent is larger than :attr:`Emax` after rounding has occurred. If not trapped, the result depends on the rounding mode, either pulling inward to the largest representable finite number or rounding outward to :const:`Infinity`. In either case, :class:`Inexact` and :class:`Rounded` are also signaled.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1605**); ***backlink***

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1605**); ***backlink***

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, **line 1605**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1605); *backlink*

Unknown interpreted text role "class".

Rounding occurred though possibly no information was lost.

Signaled whenever rounding discards digits; even if those digits are zero (such as rounding :const:`5.00` to :const:`5.0`). If not trapped, returns the result unchanged. This signal is used to detect loss of significant digits.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1616); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1616); *backlink*

Unknown interpreted text role "const".

Exponent was lower than :attr:`Emin` prior to rounding.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1624); *backlink*

Unknown interpreted text role "attr".

Occurs when an operation result is subnormal (the exponent is too small). If not trapped, returns the result unchanged.

Numerical underflow with result rounded to zero.

Occurs when a subnormal result is pushed to zero by rounding. :class:`Inexact` and :class:`Subnormal` are also signaled.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1634); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1634); *backlink*

Unknown interpreted text role "class".

Enable stricter semantics for mixing floats and Decimals.

If the signal is not trapped (default), mixing floats and Decimals is permitted in the :class:`~decimal.Decimal` constructor, :meth:`~decimal.Context.create_decimal` and all comparison operators. Both conversion and comparisons are exact. Any occurrence of a mixed operation is silently recorded by setting :exc:`FloatOperation` in the context flags. Explicit conversions with :meth:`~decimal.Decimal.from_float` or :meth:`~decimal.Context.create_decimal_from_float` do not set the flag.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1642); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1642); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1642); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1642); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1642); *backlink*

Unknown interpreted text role "meth".

Otherwise (the signal is trapped), only equality comparisons and explicit conversions are silent. All other mixed operations raise :exc:`FloatOperation`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1650); *backlink*

Unknown interpreted text role "exc".

The following table summarizes the hierarchy of signals:

```
exceptions.ArithmeticError(exceptions.Exception)
    DecimalException
        Clamped
        DivisionByZero(DecimalException, exceptions.ZeroDivisionError)
        Inexact
            Overflow(Inexact, Rounded)
            Underflow(Inexact, Rounded, Subnormal)
        InvalidOperation
        Rounded
        Subnormal
        FloatOperation(DecimalException, exceptions.TypeError)
```

Floating Point Notes

Mitigating round-off error with increased precision

The use of decimal floating point eliminates decimal representation error (making it possible to represent :const:`0.1` exactly); however, some operations can still incur round-off error when non-zero digits exceed the fixed precision.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1681); *backlink*

Unknown interpreted text role "const".

The effects of round-off error can be amplified by the addition or subtraction of nearly offsetting quantities resulting in loss of significance. Knuth provides two instructive examples where rounded floating point arithmetic with insufficient precision causes the breakdown of the associative and distributive properties of addition:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1691)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   # Examples from Seminumerical Algorithms, Section 4.2.2.
   >>> from decimal import Decimal, getcontext
   >>> getcontext().prec = 8

   >>> u, v, w = Decimal(11111113), Decimal(-11111111), Decimal('7.51111111')
   >>> (u + v) + w
   Decimal('9.5111111')
   >>> u + (v + w)
   Decimal('10')

   >>> u, v, w = Decimal(20000), Decimal(-6), Decimal('6.0000003')
   >>> (u*v) + (u*w)
   Decimal('0.01')
   >>> u * (v+w)
   Decimal('0.0060000')
```

The :mod:`decimal` module makes it possible to restore the identities by expanding the precision sufficiently to avoid loss of significance:

**System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1709); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1712)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   >>> getcontext().prec = 20
   >>> u, v, w = Decimal(11111113), Decimal(-11111111), Decimal('7.51111111')
   >>> (u + v) + w
   Decimal('9.51111111')
   >>> u + (v + w)
   Decimal('9.51111111')
   >>>
   >>> u, v, w = Decimal(20000), Decimal(-6), Decimal('6.0000003')
   >>> (u*v) + (u*w)
   Decimal('0.0060000')
   >>> u * (v+w)
   Decimal('0.0060000')
```

Special values

The number system for the :mod:`decimal` module provides special values including :const:`NaN`, :const:`sNaN`, :const:`-Infinity`, :const:`Infinity`, and two zeros, :const:`+0` and :const:`-0`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1731); *backlink*

Unknown interpreted text role "const".

Infinities can be constructed directly with: `Decimal('Infinity')`. Also, they can arise from dividing by zero when the :exc:`DivisionByZero` signal is not trapped. Likewise, when the :exc:`Overflow` signal is not trapped, infinity can result from rounding beyond the limits of the largest representable number.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1735); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1735); *backlink*

Unknown interpreted text role "exc".

The infinities are signed (affine) and can be used in arithmetic operations where they get treated as very large, indeterminate numbers. For instance, adding a constant to infinity gives another infinite result.

Some operations are indeterminate and return :const:`NaN`, or if the :exc:`InvalidOperation` signal is trapped, raise an exception. For example, `0/0` returns :const:`NaN` which means "not a number". This variety of :const:`NaN` is quiet and, once created, will flow through other computations always resulting in another :const:`NaN`. This behavior can be useful for a series of computations that occasionally have missing inputs --- it allows the calculation to proceed while flagging specific results as invalid.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1744); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1744); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1744); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1744); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1744); *backlink*

Unknown interpreted text role "const".

A variant is :const:`sNaN` which signals rather than remaining quiet after every operation. This is a useful return value when an invalid result needs to interrupt a calculation for special handling.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1752); *backlink*

Unknown interpreted text role "const".

The behavior of Python's comparison operators can be a little surprising where a :const:`NaN` is involved. A test for equality where one of the operands is a quiet or signaling :const:`NaN` always returns :const:`False` (even when doing `Decimal('NaN')==Decimal('NaN')`), while a test for inequality always returns :const:`True`. An attempt to compare two Decimals using any of the <, <=, > or >= operators will raise the :exc:`InvalidOperation` signal if either operand is a :const:`NaN`, and return :const:`False` if this signal is not trapped. Note that the General Decimal Arithmetic specification does not specify the behavior of direct comparisons; these rules for comparisons involving a :const:`NaN` were taken from the IEEE 854 standard (see Table 3 in section 5.7). To ensure strict standards-compliance, use the :meth:`compare` and :meth:`compare-signal` methods instead.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-

main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "exc".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1756); *backlink*

Unknown interpreted text role "meth".

The signed zeros can result from calculations that underflow. They keep the sign that would have resulted if the calculation had been carried out to greater precision. Since their magnitude is zero, both positive and negative zeros are treated as equal and their sign is informational.

In addition to the two signed zeros which are distinct yet equal, there are various representations of zero with differing precisions yet equivalent in value. This takes a bit of getting used to. For an eye accustomed to normalized floating point representations, it is not immediately obvious that the following calculation returns a value equal to zero:

```
>>> 1 / Decimal('Infinity')
Decimal('0E-1000026')
```

Working with threads

The :func:`getcontext` function accesses a different :class:`Context` object for each thread. Having separate thread contexts means that threads may make changes (such as `getcontext().prec=10`) without interfering with other threads.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst, line 1791); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1791); *backlink*

Unknown interpreted text role "class".

Likewise, the :func:`setcontext` function automatically assigns its target to the current thread.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1795); *backlink*

Unknown interpreted text role "func".

If :func:`setcontext` has not been called before :func:`getcontext`, then :func:`getcontext` will automatically create a new context for use in the current thread.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1798); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1798); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1798); *backlink*

Unknown interpreted text role "func".

The new context is copied from a prototype context called *DefaultContext*. To control the defaults so that each thread will use the same values throughout the application, directly modify the *DefaultContext* object. This should be done *before* any threads are started so that there won't be a race condition between threads calling :func:`getcontext`. For example:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1802); *backlink*

Unknown interpreted text role "func".

```
# Set applicationwide defaults for all threads about to be launched
DefaultContext.prec = 12
DefaultContext.rounding = ROUND_DOWN
DefaultContext.traps = ExtendedContext.traps.copy()
DefaultContext.traps[InvalidOperation] = 1
setcontext(DefaultContext)

# Afterwards, the threads can be started
t1.start()
t2.start()
t3.start()
 . . .
```

Recipes

Here are a few recipes that serve as utility functions and that demonstrate ways to work with the :class:`Decimal` class:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1829); *backlink*

Unknown interpreted text role "class".

```
def moneyfmt(value, places=2, curr='', sep=',', dp='.',
             pos='', neg='-', trailneg=''):
    """Convert Decimal to a money formatted string.

    places:  required number of places after the decimal point
    curr:    optional currency symbol before the sign (may be blank)
    sep:     optional grouping separator (comma, period, space, or blank)
    dp:      decimal point indicator (comma or period)
```

```
             only specify as blank when places is zero
    pos:     optional sign for positive numbers: '+', space or blank
    neg:     optional sign for negative numbers: '-', '(', space or blank
    trailneg:optional trailing minus indicator:  '-', ')', space or blank

    >>> d = Decimal('-1234567.8901')
    >>> moneyfmt(d, curr='$')
    '-$1,234,567.89'
    >>> moneyfmt(d, places=0, sep='.', dp='', neg='', trailneg='-')
    '1.234.568-'
    >>> moneyfmt(d, curr='$', neg='(', trailneg=')')
    '($1,234,567.89)'
    >>> moneyfmt(Decimal(123456789), sep=' ')
    '123 456 789.00'
    >>> moneyfmt(Decimal('-0.02'), neg='<', trailneg='>')
    '<0.02>'

    """
    q = Decimal(10) ** -places      # 2 places --> '0.01'
    sign, digits, exp = value.quantize(q).as_tuple()
    result = []
    digits = list(map(str, digits))
    build, next = result.append, digits.pop
    if sign:
        build(trailneg)
    for i in range(places):
        build(next() if digits else '0')
    if places:
        build(dp)
    if not digits:
        build('0')
    i = 0
    while digits:
        build(next())
        i += 1
        if i == 3 and digits:
            i = 0
            build(sep)
    build(curr)
    build(neg if sign else pos)
    return ''.join(reversed(result))

def pi():
    """Compute Pi to the current precision.

    >>> print(pi())
    3.141592653589793238462643383

    """
    getcontext().prec += 2  # extra digits for intermediate steps
    three = Decimal(3)      # substitute "three=3.0" for regular floats
    lasts, t, s, n, na, d, da = 0, three, 3, 1, 0, 0, 24
    while s != lasts:
        lasts = s
        n, na = n+na, na+8
        d, da = d+da, da+32
        t = (t * n) / d
        s += t
    getcontext().prec -= 2
    return +s               # unary plus applies the new precision

def exp(x):
    """Return e raised to the power of x.  Result type matches input type.

    >>> print(exp(Decimal(1)))
    2.718281828459045235360287471
    >>> print(exp(Decimal(2)))
    7.389056098930650227230427461
    >>> print(exp(2.0))
    7.38905609893
    >>> print(exp(2+0j))
    (7.38905609893+0j)

    """
    getcontext().prec += 2
    i, lasts, s, fact, num = 0, 0, 1, 1, 1
    while s != lasts:
        lasts = s
        i += 1
        fact *= i
        num *= x
```

```
            s += num / fact
        getcontext().prec -= 2
        return +s

    def cos(x):
        """Return the cosine of x as measured in radians.

        The Taylor series approximation works best for a small value of x.
        For larger values, first compute x = x % (2 * pi).

        >>> print(cos(Decimal('0.5')))
        0.8775825618903727161162815826
        >>> print(cos(0.5))
        0.87758256189
        >>> print(cos(0.5+0j))
        (0.87758256189+0j)

        """
        getcontext().prec += 2
        i, lasts, s, fact, num, sign = 0, 0, 1, 1, 1, 1
        while s != lasts:
            lasts = s
            i += 2
            fact *= i * (i-1)
            num *= x * x
            sign *= -1
            s += num / fact * sign
        getcontext().prec -= 2
        return +s

    def sin(x):
        """Return the sine of x as measured in radians.

        The Taylor series approximation works best for a small value of x.
        For larger values, first compute x = x % (2 * pi).

        >>> print(sin(Decimal('0.5')))
        0.4794255386042030002732879352
        >>> print(sin(0.5))
        0.479425538604
        >>> print(sin(0.5+0j))
        (0.479425538604+0j)

        """
        getcontext().prec += 2
        i, lasts, s, fact, num, sign = 1, 0, x, 1, x, 1
        while s != lasts:
            lasts = s
            i += 2
            fact *= i * (i-1)
            num *= x * x
            sign *= -1
            s += num / fact * sign
        getcontext().prec -= 2
        return +s
```

Decimal FAQ

Q. It is cumbersome to type `decimal.Decimal('1234.5')`. Is there a way to minimize typing when using the interactive interpreter?

A. Some users abbreviate the constructor to just a single letter:

```
>>> D = decimal.Decimal
>>> D('1.23') + D('3.45')
Decimal('4.68')
```

Q. In a fixed-point application with two decimal places, some inputs have many places and need to be rounded. Others are not supposed to have excess digits and need to be validated. What methods should be used?

A. The :meth:`quantize` method rounds to a fixed number of decimal places. If the :const:`Inexact` trap is set, it is also useful for validation:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1999); *backlink*

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 1999); *backlink*

Unknown interpreted text role "const".

```
>>> TWOPLACES = Decimal(10) ** -2       # same as Decimal('0.01')
```

```
>>> # Round to two places
>>> Decimal('3.214').quantize(TWOPLACES)
Decimal('3.21')
```

```
>>> # Validate that a number does not exceed two places
>>> Decimal('3.21').quantize(TWOPLACES, context=Context(traps=[Inexact]))
Decimal('3.21')
```

```
>>> Decimal('3.214').quantize(TWOPLACES, context=Context(traps=[Inexact]))
Traceback (most recent call last):
   ...
Inexact: None
```

Q. Once I have valid two place inputs, how do I maintain that invariant throughout an application?

A. Some operations like addition, subtraction, and multiplication by an integer will automatically preserve fixed point. Others operations, like division and non-integer multiplication, will change the number of decimal places and need to be followed-up with a :meth:`quantize` step:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2020); *backlink*

Unknown interpreted text role "meth".

```
>>> a = Decimal('102.72')           # Initial fixed-point values
>>> b = Decimal('3.17')
>>> a + b                           # Addition preserves fixed-point
Decimal('105.89')
>>> a - b
Decimal('99.55')
>>> a * 42                          # So does integer multiplication
Decimal('4314.24')
>>> (a * b).quantize(TWOPLACES)     # Must quantize non-integer multiplication
Decimal('325.62')
>>> (b / a).quantize(TWOPLACES)     # And quantize division
Decimal('0.03')
```

In developing fixed-point applications, it is convenient to define functions to handle the :meth:`quantize` step:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2038); *backlink*

Unknown interpreted text role "meth".

```
>>> def mul(x, y, fp=TWOPLACES):
...     return (x * y).quantize(fp)
>>> def div(x, y, fp=TWOPLACES):
...     return (x / y).quantize(fp)
```

```
>>> mul(a, b)                       # Automatically preserve fixed-point
Decimal('325.62')
>>> div(b, a)
Decimal('0.03')
```

Q. There are many ways to express the same value. The numbers :const:`200`, :const:`200.000`, :const:`2E2`, and :const:`.02E+4` all have the same value at various precisions. Is there a way to transform them to a single recognizable canonical value?

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2051); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2051); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2051); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2051); *backlink*

Unknown interpreted text role "const".

A. The :meth:`normalize` method maps all equivalent values to a single representative:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2056); *backlink*

Unknown interpreted text role "meth".

```
>>> values = map(Decimal, '200 200.000 2E2 .02E+4'.split())
>>> [v.normalize() for v in values]
[Decimal('2E+2'), Decimal('2E+2'), Decimal('2E+2'), Decimal('2E+2')]
```

Q. Some decimal values always print with exponential notation. Is there a way to get a non-exponential representation?

A. For some values, exponential notation is the only way to express the number of significant places in the coefficient. For example, expressing :const:`5.0E+3` as :const:`5000` keeps the value constant but cannot show the original's two-place significance.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2066); *backlink*

Unknown interpreted text role "const".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2066); *backlink*

Unknown interpreted text role "const".

If an application does not care about tracking significance, it is easy to remove the exponent and trailing zeroes, losing significance, but keeping the value unchanged:

```
>>> def remove_exponent(d):
...     return d.quantize(Decimal(1)) if d == d.to_integral() else d.normalize()
```

```
>>> remove_exponent(Decimal('5E+3'))
Decimal('5000')
```

Q. Is there a way to convert a regular float to a :class:`Decimal`?

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2081); *backlink*

Unknown interpreted text role "class".

A. Yes, any binary floating point number can be exactly expressed as a Decimal though an exact conversion may take more precision than intuition would suggest:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2087)

Unknown directive type "doctest".

```
.. doctest::

    >>> Decimal(math.pi)
    Decimal('3.141592653589793115997963468544185161590576171875')
```

Q. Within a complex calculation, how can I make sure that I haven't gotten a spurious result because of insufficient precision or rounding anomalies.

A. The decimal module makes it easy to test results. A best practice is to re-run calculations using greater precision and with various rounding modes. Widely differing results indicate insufficient precision, rounding mode issues, ill-conditioned inputs, or a numerically unstable algorithm.

Q. I noticed that context precision is applied to the results of operations but not to the inputs. Is there anything to watch out for when mixing values of different precisions?

A. Yes. The principle is that all values are considered to be exact and so is the arithmetic on those values. Only the results are rounded. The advantage for inputs is that "what you type is what you get". A disadvantage is that the results can look odd if you forget that the inputs haven't been rounded:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2109)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   >>> getcontext().prec = 3
   >>> Decimal('3.104') + Decimal('2.104')
   Decimal('5.21')
   >>> Decimal('3.104') + Decimal('0.000') + Decimal('2.104')
   Decimal('5.20')
```

The solution is either to increase precision or to force rounding of inputs using the unary plus operation:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2120)

Unknown directive type "doctest".

```
.. doctest:: newcontext

   >>> getcontext().prec = 3
   >>> +Decimal('1.23456789')      # unary plus triggers rounding
   Decimal('1.23')
```

Alternatively, inputs can be rounded upon creation using the :meth:`Context.create_decimal` method:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2126); *backlink*

Unknown interpreted text role "meth".

```
>>> Context(prec=5, rounding=ROUND_DOWN).create_decimal('1.2345678')
Decimal('1.2345')
```

Q. Is the CPython implementation fast for large numbers?

A. Yes. In the CPython and PyPy3 implementations, the C/CFFI versions of the decimal module integrate the high speed libmpdec library for arbitrary precision correctly-rounded decimal floating point arithmetic [1]. `libmpdec` uses Karatsuba multiplication for medium-sized numbers and the Number Theoretic Transform for very large numbers.

The context must be adapted for exact arbitrary precision arithmetic. :attr:`Emin` and :attr:`Emax` should always be set to the maximum values, :attr:`clamp` should always be 0 (the default). Setting :attr:`prec` requires some care.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2144); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2144); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2144); *backlink*

Unknown interpreted text role "attr".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2144); *backlink*

Unknown interpreted text role "attr".

The easiest approach for trying out bignum arithmetic is to use the maximum value for :attr:`prec` as well [2]:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2148); *backlink*

Unknown interpreted text role "attr".

```
>>> setcontext(Context(prec=MAX_PREC, Emax=MAX_EMAX, Emin=MIN_EMIN))
>>> x = Decimal(2) ** 256
>>> x / 128
Decimal('904625697166532776746648320380374280103671755200316906558262375061821325312')
```

For inexact results, :attr:`MAX_PREC` is far too large on 64-bit platforms and the available memory will be insufficient:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2157); *backlink*

Unknown interpreted text role "attr".

```
>>> Decimal(1) / 3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
MemoryError
```

On systems with overallocation (e.g. Linux), a more sophisticated approach is to adjust :attr:`prec` to the amount of available RAM. Suppose that you have 8GB of RAM and expect 10 simultaneous operands using a maximum of 500MB each:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2165); *backlink*

Unknown interpreted text role "attr".

```
>>> import sys
>>>
>>> # Maximum number of digits for a single operand using 500MB in 8-byte words
>>> # with 19 digits per word (4-byte and 9 digits for the 32-bit build):
>>> maxdigits = 19 * ((500 * 1024**2) // 8)
>>>
>>> # Check that this works:
>>> c = Context(prec=maxdigits, Emax=MAX_EMAX, Emin=MIN_EMIN)
>>> c.traps[Inexact] = True
>>> setcontext(c)
>>>
>>> # Fill the available precision with nines:
>>> x = Decimal(0).logical_invert() * 9
>>> sys.getsizeof(x)
524288112
>>> x + 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  decimal.Inexact: [<class 'decimal.Inexact'>]
```

In general (and especially on systems without overallocation), it is recommended to estimate even tighter bounds and set the :attr:`Inexact` trap if all calculations are expected to be exact.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2189); *backlink*

Unknown interpreted text role "attr".

[1] **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2195)

Unknown directive type "versionadded".

```
.. versionadded:: 3.3
```

[2]

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]decimal.rst`, line 2198)

Unknown directive type "versionchanged".

```
.. versionchanged:: 3.9
   This approach now works for all exact results except for non-integer powers.
```