class **float**(object)

| float(x) -> floating point number

|

| Convert a string or number to a floating point number,

if possible.

|

| Methods defined here:

|

| \_\_abs\_\_(...)

| x.\_\_abs\_\_() <==> abs(x)

|

| \_\_add\_\_(...)

| x.\_\_add\_\_(y) <==> x+y

|

| \_\_coerce\_\_(...)

| x.\_\_coerce\_\_(y) <==> coerce(x, y)

|

| \_\_div\_\_(...)

| x.\_\_div\_\_(y) <==> x/y

|

| \_\_divmod\_\_(...)

| x.\_\_divmod\_\_(y) <==> divmod(x, y)

|

| \_\_eq\_\_(...)

| x.\_\_eq\_\_(y) <==> x==y

|

| \_\_float\_\_(...)

| x.\_\_float\_\_() <==> float(x)

|

| \_\_floordiv\_\_(...)

| x.\_\_floordiv\_\_(y) <==> x//y

|

| \_\_format\_\_(...)

| float.\_\_format\_\_(format\_spec) -> string

|

| Formats the float according to format\_spec.

|

| \_\_ge\_\_(...)

| x.\_\_ge\_\_(y) <==> x>=y

|

| \_\_getattribute\_\_(...)

| x.\_\_getattribute\_\_('name') <==> x.name

|

| \_\_getformat\_\_(...)

| float.\_\_getformat\_\_(typestr) -> string

|

| You probably don't want to use this function. It

exists mainly to be

| used in Python's test suite.

|

| typestr must be 'double' or 'float'. This function

returns whichever of

| 'unknown', 'IEEE, big-endian' or 'IEEE, little-

endian' best describes the

| format of floating point numbers used by the C type

named by typestr.

|

| \_\_getnewargs\_\_(...)

|

| \_\_gt\_\_(...)

| x.\_\_gt\_\_(y) <==> x>y

|

| \_\_hash\_\_(...)

| x.\_\_hash\_\_() <==> hash(x)

|

| \_\_int\_\_(...)

| x.\_\_int\_\_() <==> int(x)

|

| \_\_le\_\_(...)

| x.\_\_le\_\_(y) <==> x<=y

|

| \_\_long\_\_(...)

| x.\_\_long\_\_() <==> long(x)

|

| \_\_lt\_\_(...)

| x.\_\_lt\_\_(y) <==> x<y

|

| \_\_mod\_\_(...)

| x.\_\_mod\_\_(y) <==> x%y

|

| \_\_mul\_\_(...)

| x.\_\_mul\_\_(y) <==> x\*y

|

| \_\_ne\_\_(...)

| x.\_\_ne\_\_(y) <==> x!=y

|

| \_\_neg\_\_(...)

| x.\_\_neg\_\_() <==> -x

|

| \_\_nonzero\_\_(...)

| x.\_\_nonzero\_\_() <==> x != 0

|

| \_\_pos\_\_(...)

| x.\_\_pos\_\_() <==> +x

|

| \_\_pow\_\_(...)

| x.\_\_pow\_\_(y[, z]) <==> pow(x, y[, z])

|

| \_\_radd\_\_(...)

| x.\_\_radd\_\_(y) <==> y+x

|

| \_\_rdiv\_\_(...)

| x.\_\_rdiv\_\_(y) <==> y/x

|

| \_\_rdivmod\_\_(...)

| x.\_\_rdivmod\_\_(y) <==> divmod(y, x)

|

| \_\_repr\_\_(...)

| x.\_\_repr\_\_() <==> repr(x)

|

| \_\_rfloordiv\_\_(...)

| x.\_\_rfloordiv\_\_(y) <==> y//x

|

| \_\_rmod\_\_(...)

| x.\_\_rmod\_\_(y) <==> y%x

|

| \_\_rmul\_\_(...)

| x.\_\_rmul\_\_(y) <==> y\*x

|

| \_\_rpow\_\_(...)

| y.\_\_rpow\_\_(x[, z]) <==> pow(x, y[, z])

|

| \_\_rsub\_\_(...)

| x.\_\_rsub\_\_(y) <==> y-x

|

| \_\_rtruediv\_\_(...)

| x.\_\_rtruediv\_\_(y) <==> y/x

|

| \_\_setformat\_\_(...)

| float.\_\_setformat\_\_(typestr, fmt) -> None

|

| You probably don't want to use this function. It

exists mainly to be

| used in Python's test suite.

|

| typestr must be 'double' or 'float'. fmt must be

one of 'unknown',

| 'IEEE, big-endian' or 'IEEE, little-endian', and in

addition can only be

| one of the latter two if it appears to match the

underlying C reality.

|

| Override the automatic determination of C-level

floating point type.

| This affects how floats are converted to and from

binary strings.

|

| \_\_str\_\_(...)

| x.\_\_str\_\_() <==> str(x)

|

| \_\_sub\_\_(...)

| x.\_\_sub\_\_(y) <==> x-y

|

| \_\_truediv\_\_(...)

| x.\_\_truediv\_\_(y) <==> x/y

|

| \_\_trunc\_\_(...)

| Return the Integral closest to x between 0 and x.

|

| as\_integer\_ratio(...)

| float.as\_integer\_ratio() -> (int, int)

|

| Return a pair of integers, whose ratio is exactly

equal to the original

| float and with a positive denominator.

| Raise OverflowError on infinities and a ValueError

on NaNs.

|

| >>> (10.0).as\_integer\_ratio()

| (10, 1)

| >>> (0.0).as\_integer\_ratio()

| (0, 1)

| >>> (-.25).as\_integer\_ratio()

| (-1, 4)

|

| conjugate(...)

| Return self, the complex conjugate of any float.

|

| fromhex(...)

| float.fromhex(string) -> float

|

| Create a floating-point number from a hexadecimal

string.

| >>> float.fromhex('0x1.ffffp10')

| 2047.984375

| >>> float.fromhex('-0x1p-1074')

| -4.9406564584124654e-324

|

| hex(...)

| float.hex() -> string

|

| Return a hexadecimal representation of a floating-

point number.

| >>> (-0.1).hex()

| '-0x1.999999999999ap-4'

| >>> 3.14159.hex()

| '0x1.921f9f01b866ep+1'

|

| is\_integer(...)

| Return True if the float is an integer.

|

| --------------------------------------------------------

| Data descriptors defined here:

|

| imag

| the imaginary part of a complex number

|

| real

| the real part of a complex number

|

| --------------------------------------------------------