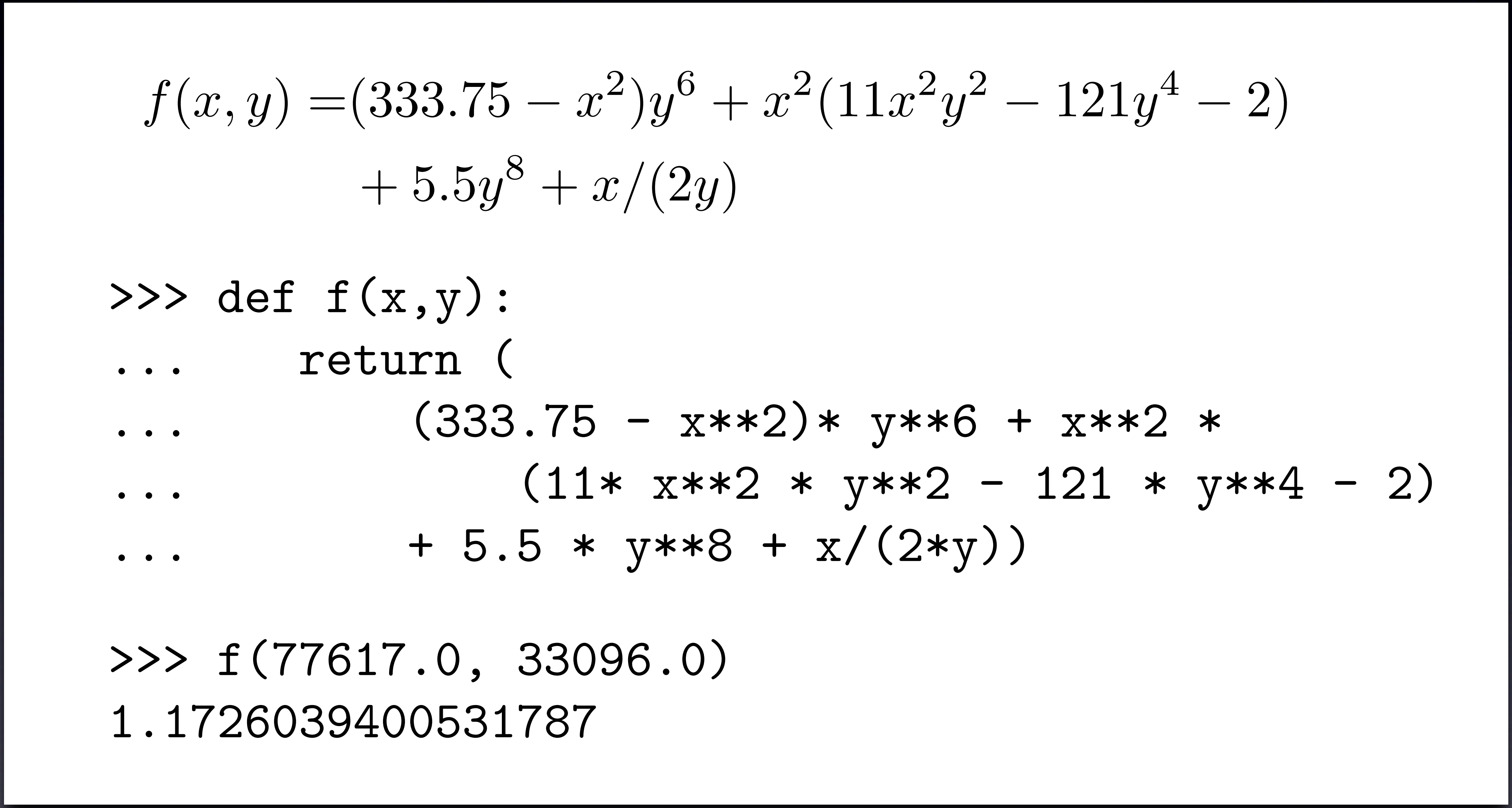
Interval arithmetic: Python implementation and applications

Stefano Taschini
Altis Investment Management AG
Switzerland
http://www.altis.ch/

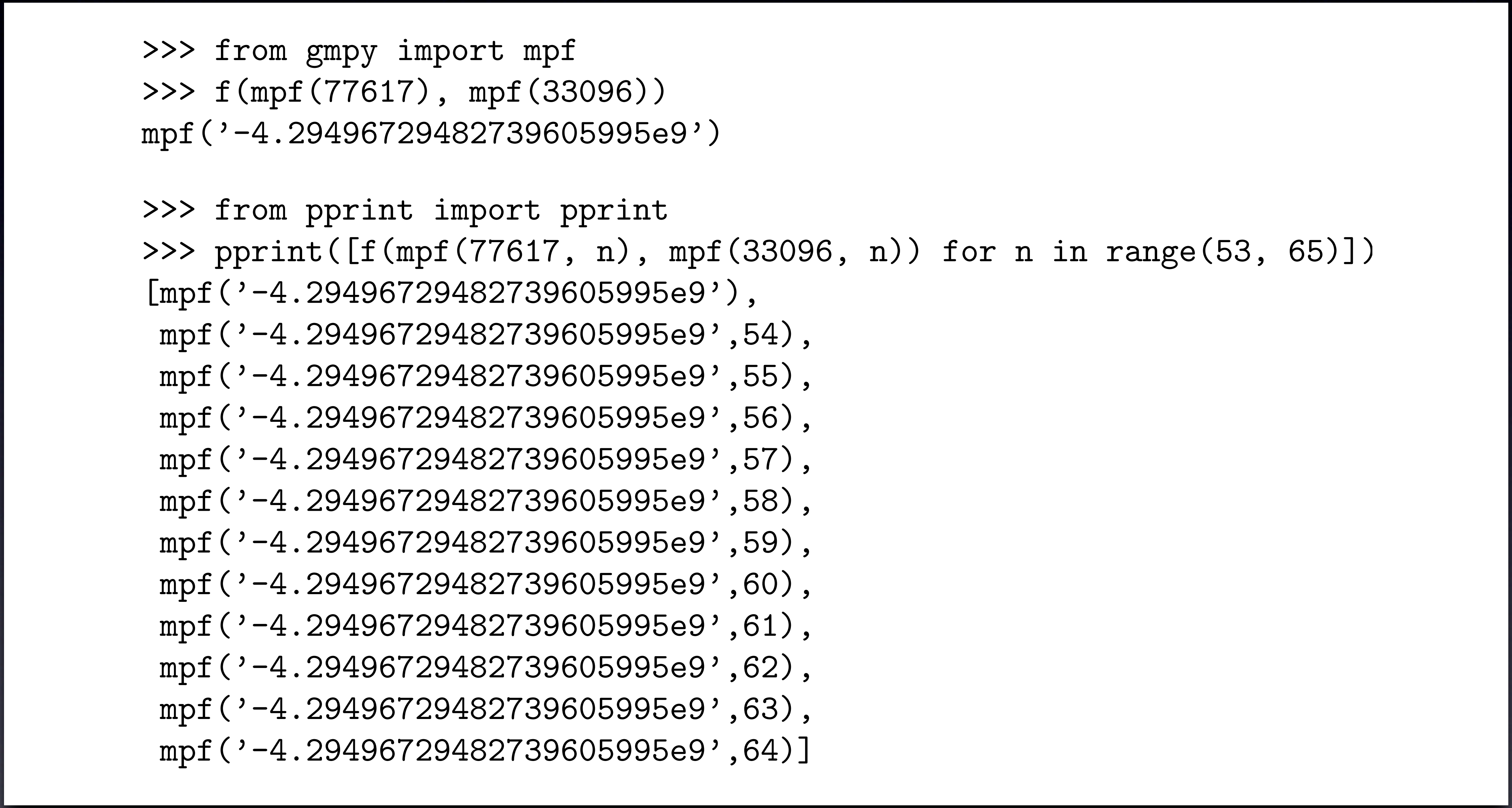
An example (1)



$$f(x,y) = (333.75 - x^2)y^6 + x^2(11x^2y^2 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

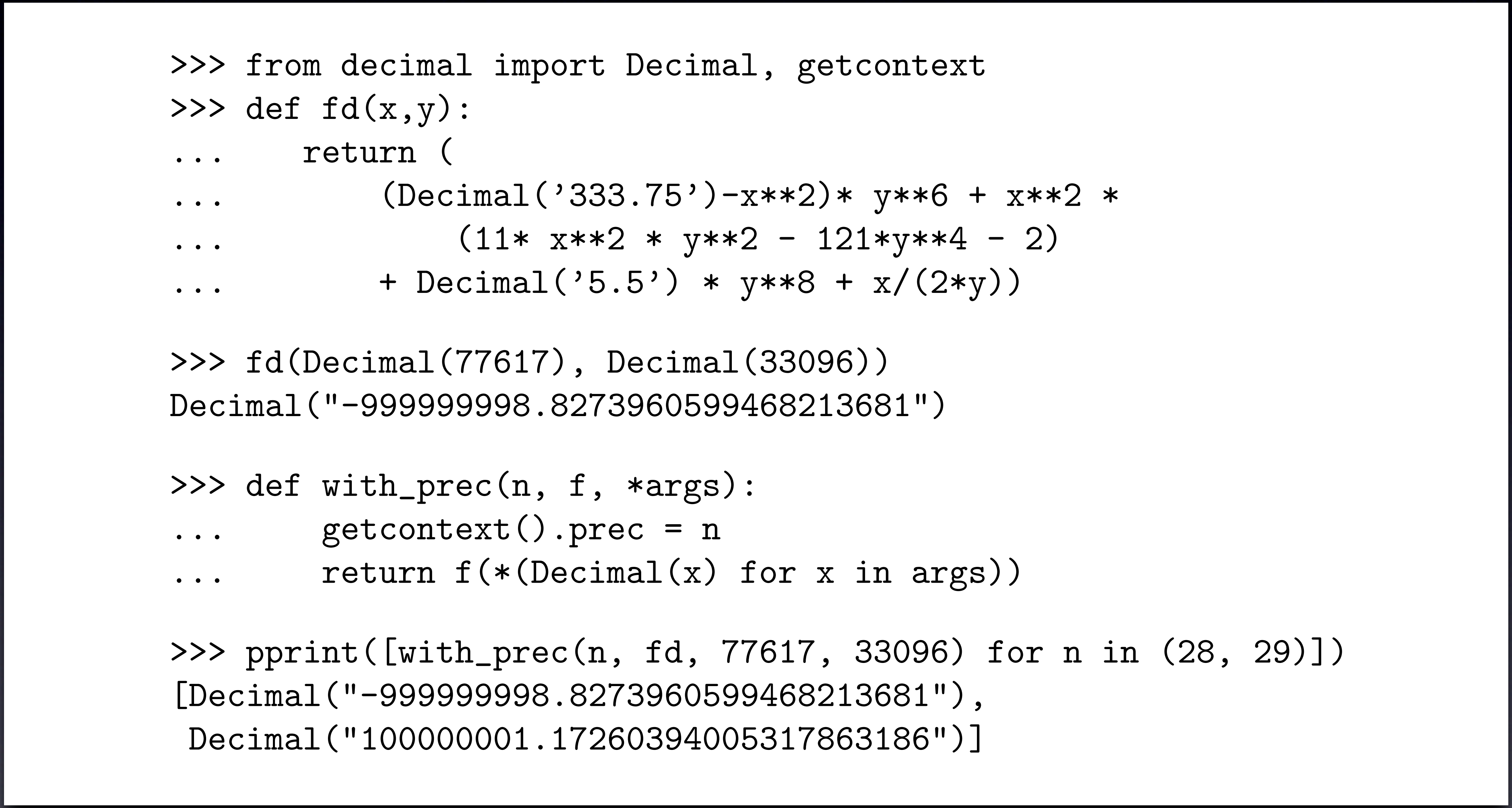
```
>>> def f(x,y):
...     return (
...         (333.75 - x**2)* y**6 + x**2 *
...             (11* x**2 * y**2 - 121 * y**4 - 2)
...         + 5.5 * y**8 + x/(2*y))

>>> f(77617.0, 33096.0)
1.1726039400531787
```

An example (2)



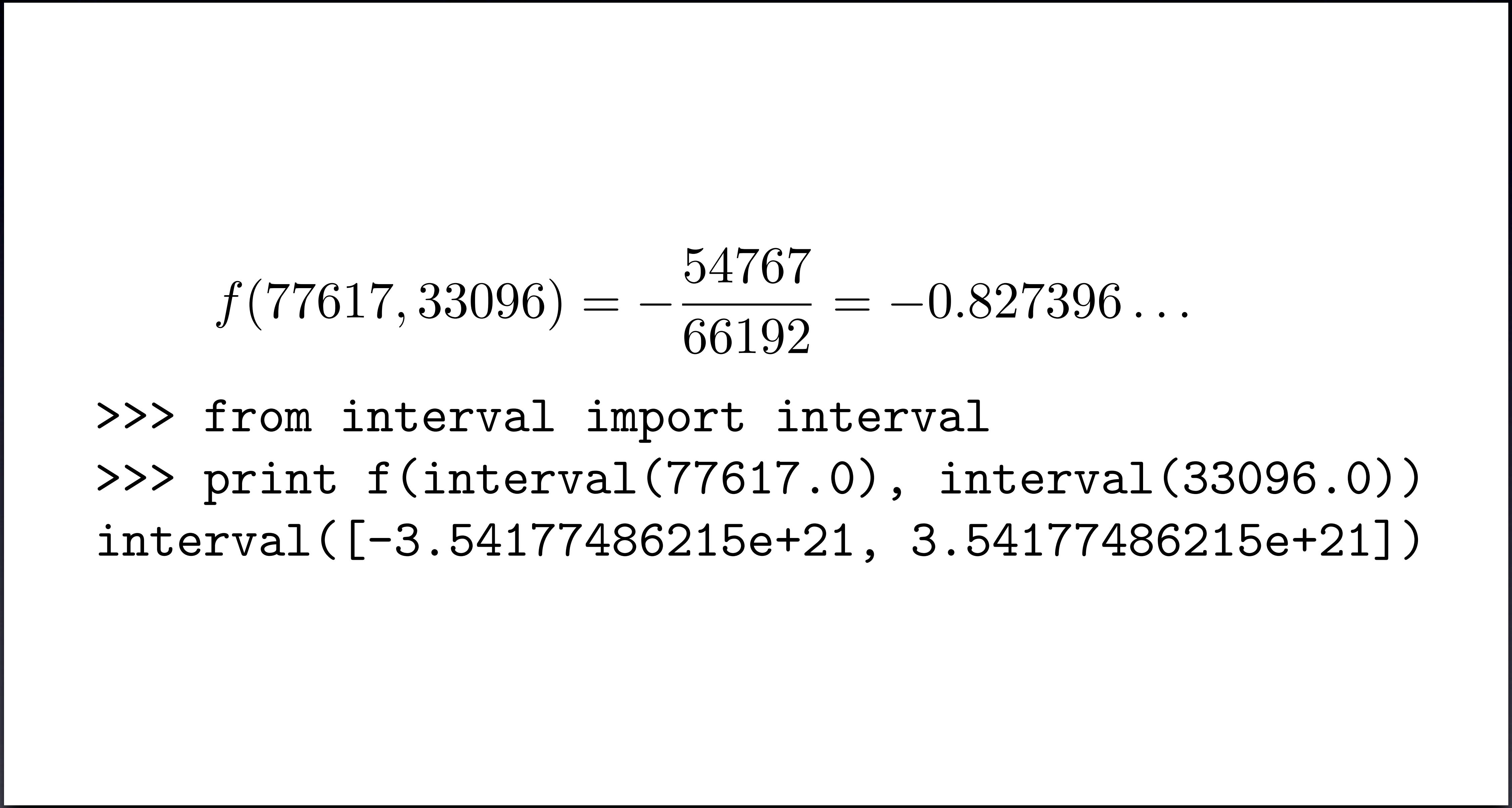
```
>>> from gmpy import mpf
>>> f(mpf(77617), mpf(33096))
mpf(’-4.29496729482739605995e9’)

>>> from pprint import pprint
>>> pprint([f(mpf(77617, n), mpf(33096, n)) for n in range(53, 65)])
[mpf(’-4.29496729482739605995e9’),
 mpf(’-4.29496729482739605995e9’,54),
 mpf(’-4.29496729482739605995e9’,55),
 mpf(’-4.29496729482739605995e9’,56),
 mpf(’-4.29496729482739605995e9’,57),
 mpf(’-4.29496729482739605995e9’,58),
 mpf(’-4.29496729482739605995e9’,59),
 mpf(’-4.29496729482739605995e9’,60),
 mpf(’-4.29496729482739605995e9’,61),
 mpf(’-4.29496729482739605995e9’,62),
 mpf(’-4.29496729482739605995e9’,63),
 mpf(’-4.29496729482739605995e9’,64)]
```

An example (3)



```
>>> from decimal import Decimal, getcontext
>>> def fd(x,y):
...    return (
...        (Decimal(’333.75’)-x**2)* y**6 + x**2 *
...            (11* x**2 * y**2 - 121*y**4 - 2)
...        + Decimal(’5.5’) * y**8 + x/(2*y))

>>> fd(Decimal(77617), Decimal(33096))
Decimal("-999999998.8273960599468213681")

>>> def with_prec(n, f, *args):
...     getcontext().prec = n
...     return f(*(Decimal(x) for x in args))

>>> pprint([with_prec(n, fd, 77617, 33096) for n in (28, 29)])
[Decimal("-999999998.8273960599468213681"),
 Decimal("100000001.17260394005317863186")]
```

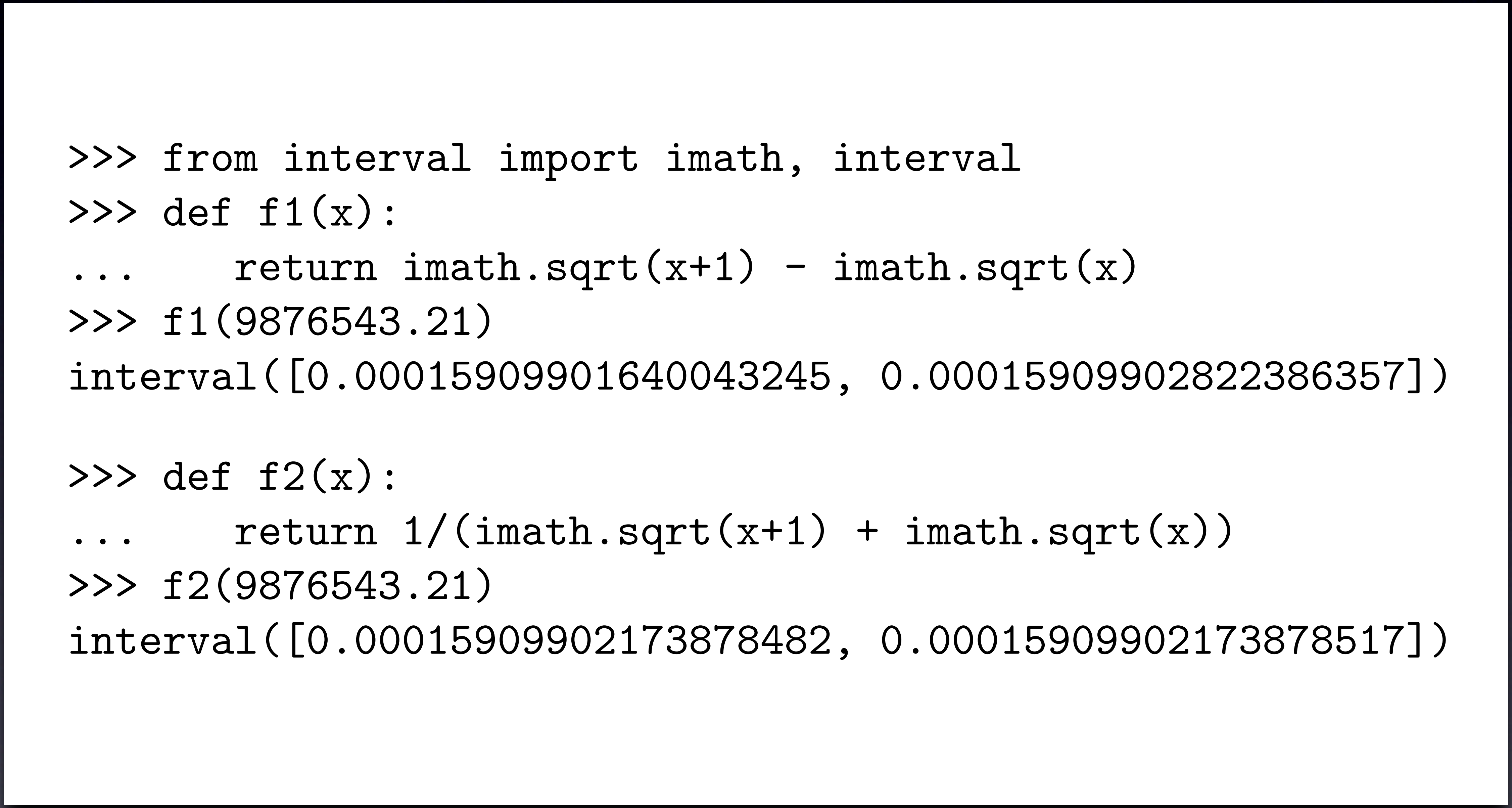
An example (4)



$$f(77617, 33096) = -\frac{54767}{66192} = -0.827396\ldots$$

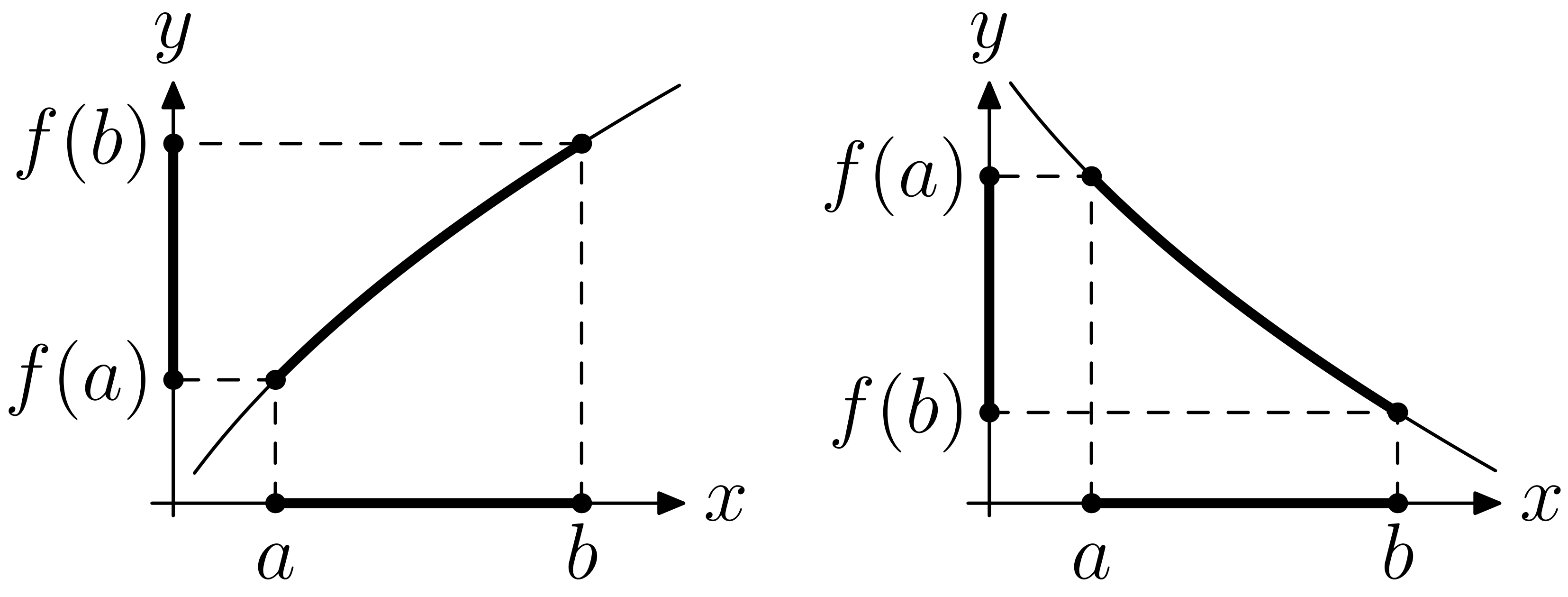
```
>>> from interval import interval
>>> print f(interval(77617.0), interval(33096.0))
interval([-3.54177486215e+21, 3.54177486215e+21])
```

Martelli’s example



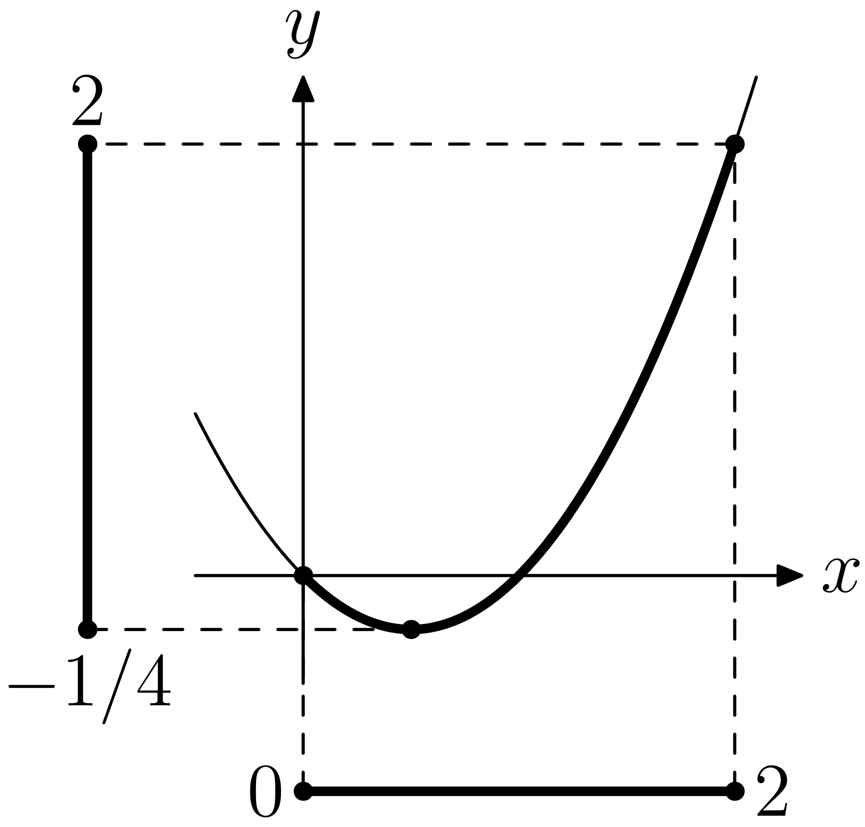
```
>>> from interval import imath, interval
>>> def f1(x):
...     return imath.sqrt(x+1) - imath.sqrt(x)
>>> f1(9876543.21)
interval([0.00015909901640043245, 0.00015909902822386357])

>>> def f2(x):
...     return 1/(imath.sqrt(x+1) + imath.sqrt(x))
>>> f2(9876543.21)
interval([0.00015909902173878482, 0.00015909902173878517])
```

Functions of intervals (1)

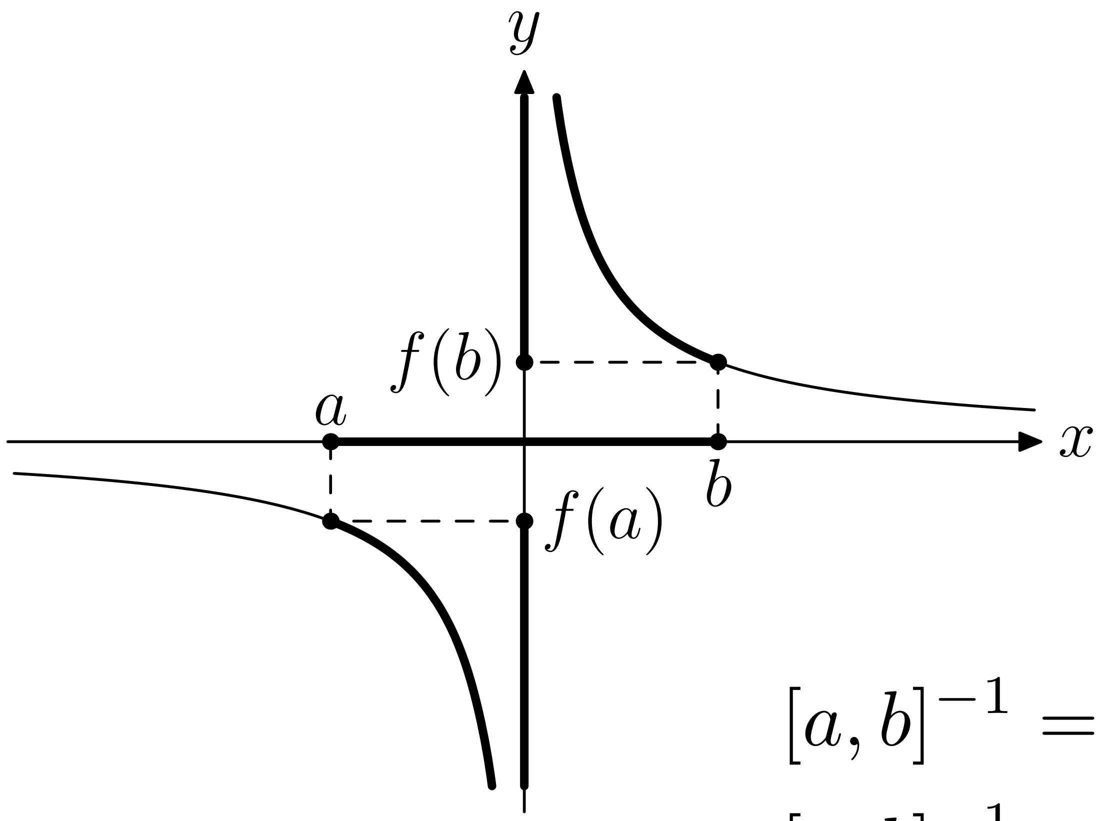


Functions of intervals (2)

$$f([a,b]) = \left[\min_{x\in[a,b]} f(x), \max_{x\in[a,b]} f(x)\right]$$



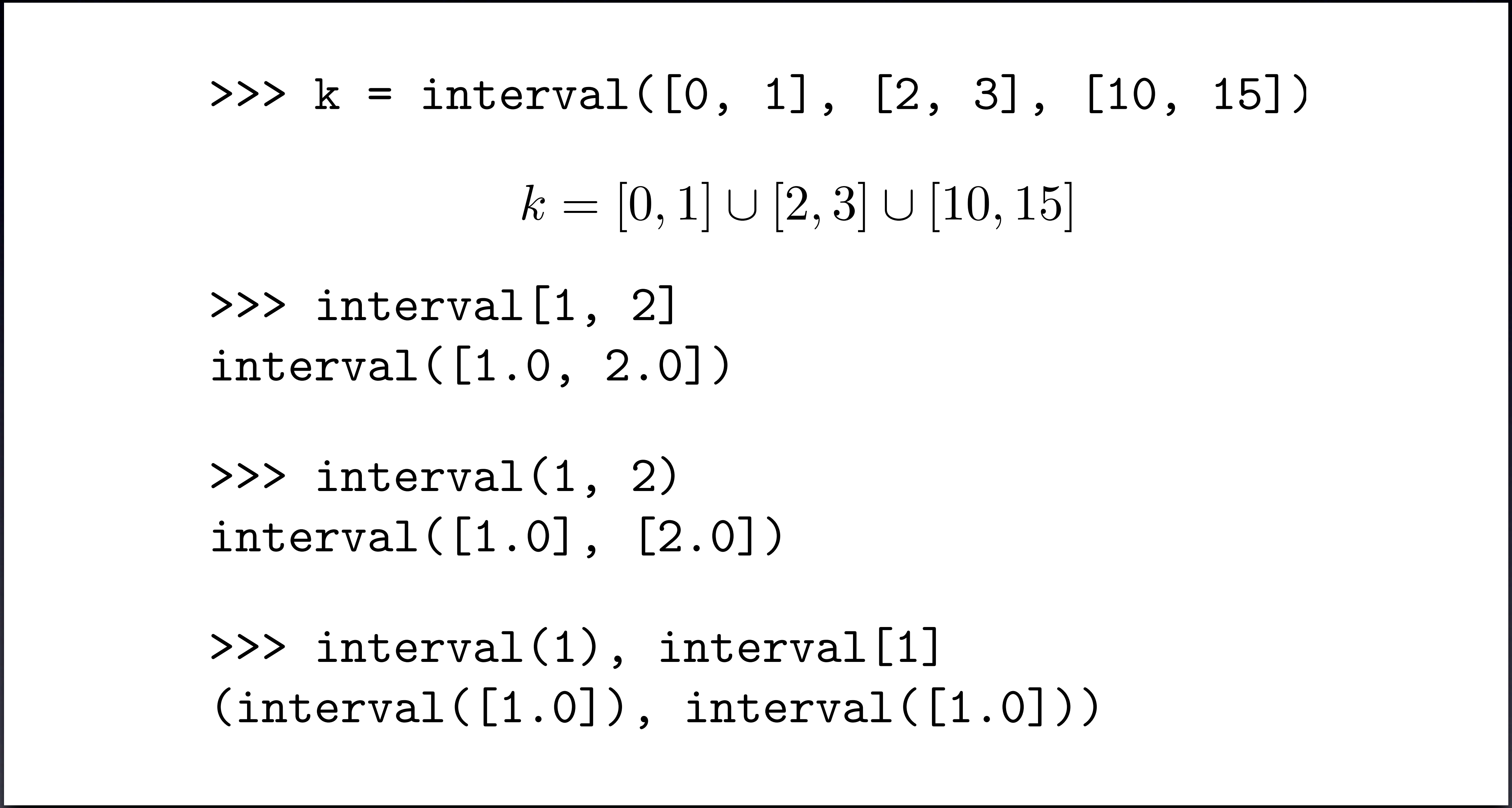
Functions of intervals (3)



$$[a, b]^{-1} = [-\infty, +\infty] \quad ?$$

$$[a, b]^{-1} = [-\infty, a^{-1}] \cup [b^{-1}, +\infty] \quad ?$$

Intervals in Python (1)



```
>>> k = interval([0, 1], [2, 3], [10, 15])
```

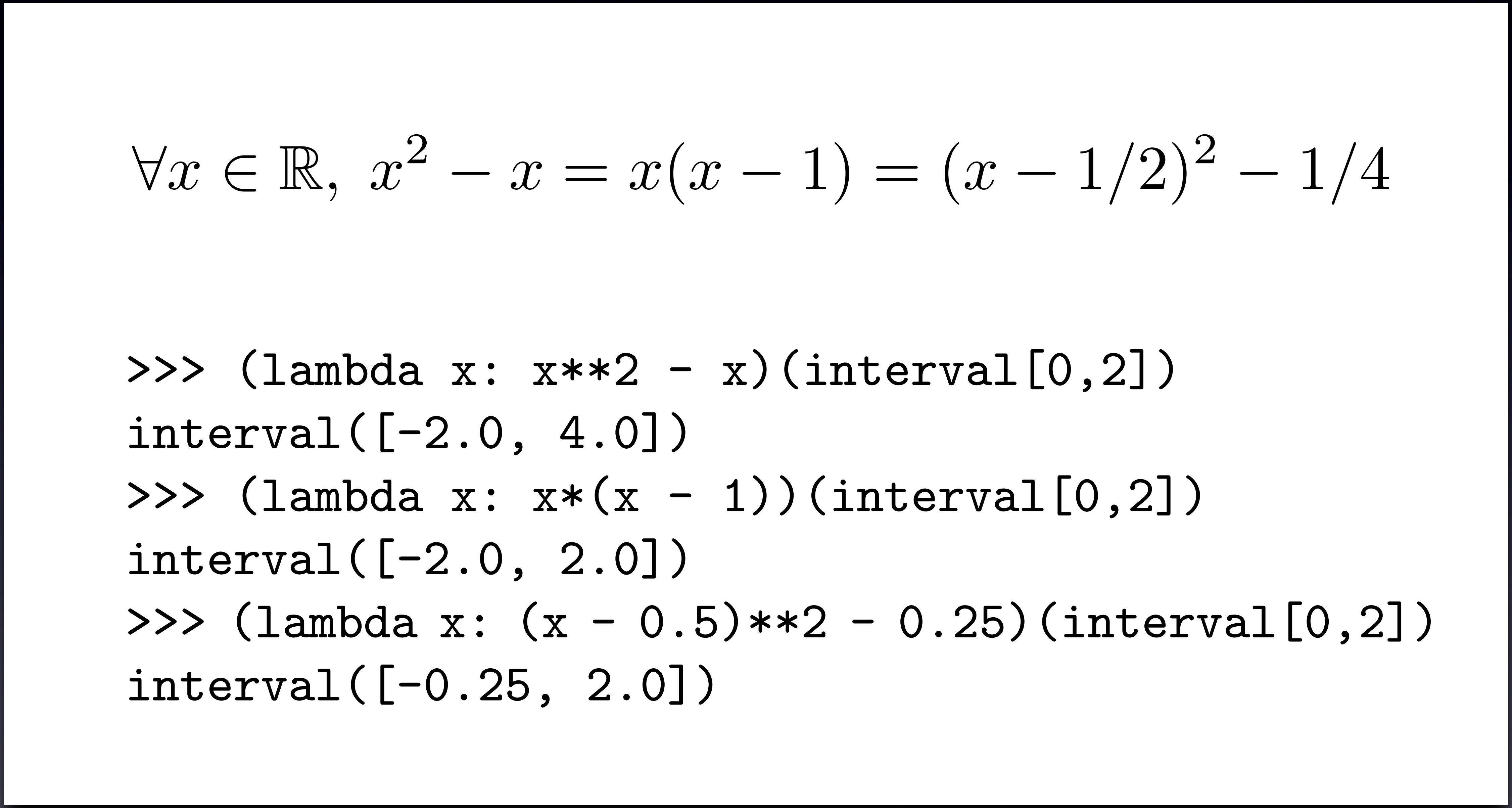
$$k = [0, 1] \cup [2, 3] \cup [10, 15]$$

```
>>> interval[1, 2]
interval([1.0, 2.0])

>>> interval(1, 2)
interval([1.0], [2.0])

>>> interval(1), interval[1]
(interval([1.0]), interval([1.0]))
```

Intervals in Python (2)

```
>>> interval[-2,+4].inverse()
interval([-inf, -0.5], [0.25, inf])

>>> interval[10] / interval[3]
interval([3.333333333333333, 3.3333333333333339])

>>> imath.exp(1)
interval([2.7182818284590451, 2.7182818284590455])

>>> imath.log(interval[-1, 1])
interval([-inf, 0.0])

>>> print imath.tanpi(interval[0.25, 0.75])
interval([-inf, -1.0], [1.0, inf])
```

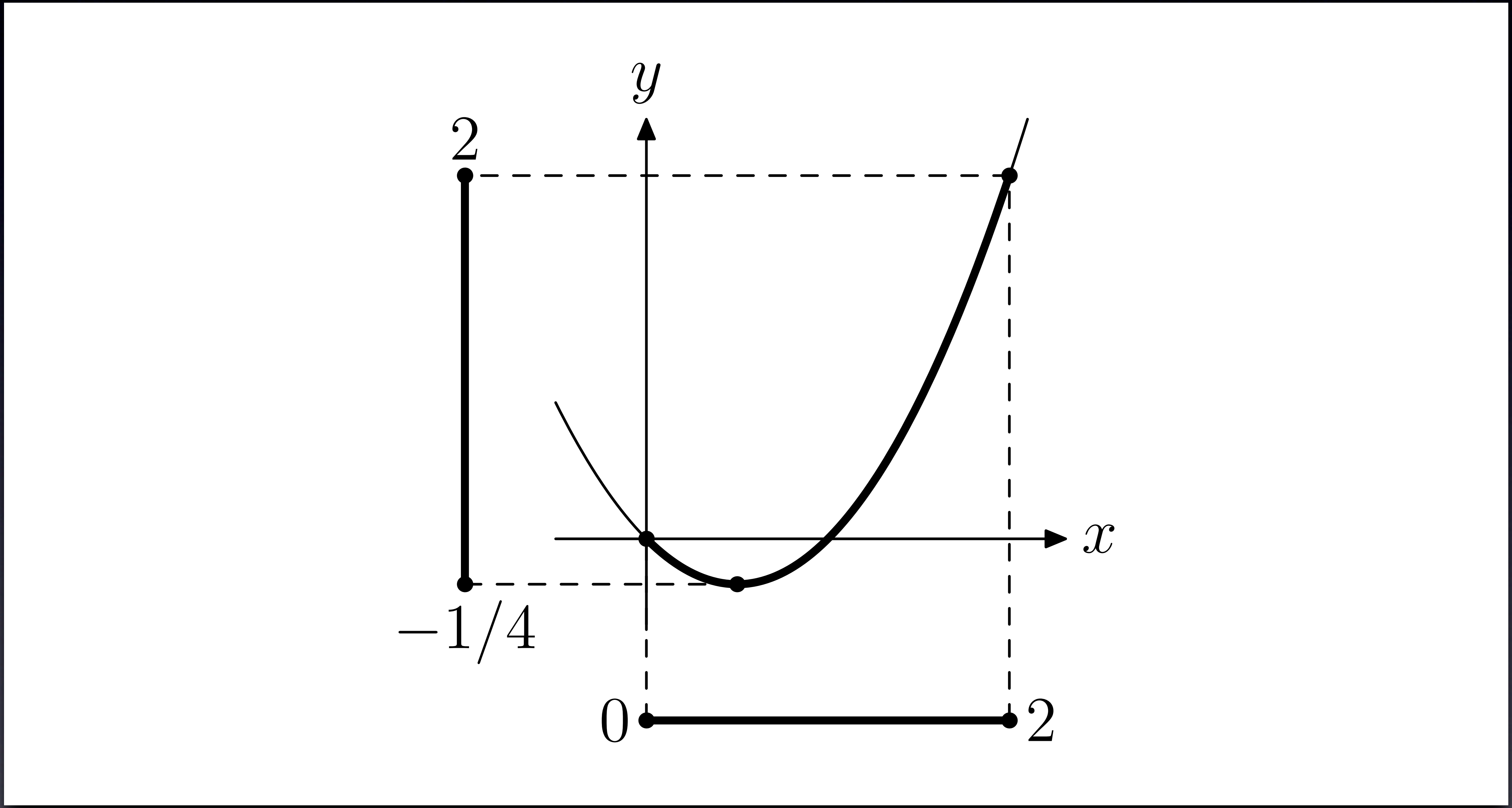
Dependency (1)



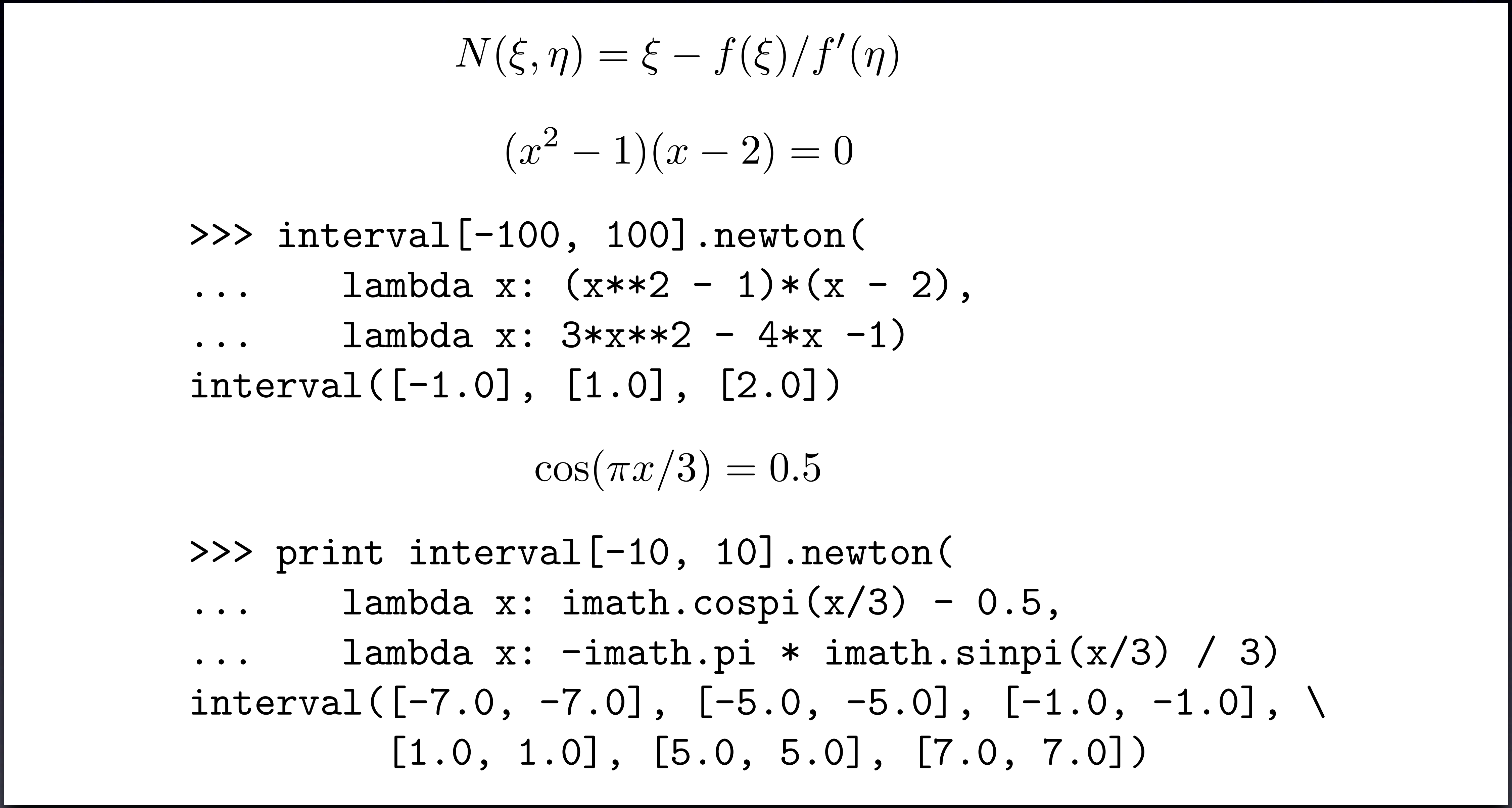
$$\forall x \in \mathbb{R},\ x^2 - x = x(x-1) = (x - 1/2)^2 - 1/4$$

```
>>> (lambda x: x**2 - x)(interval[0,2])
interval([-2.0, 4.0])
>>> (lambda x: x*(x - 1))(interval[0,2])
interval([-2.0, 2.0])
>>> (lambda x: (x - 0.5)**2 - 0.25)(interval[0,2])
interval([-0.25, 2.0])
```

Dependency (2)



Newton (1)



$$N(\xi, \eta) = \xi - f(\xi)/f'(\eta)$$

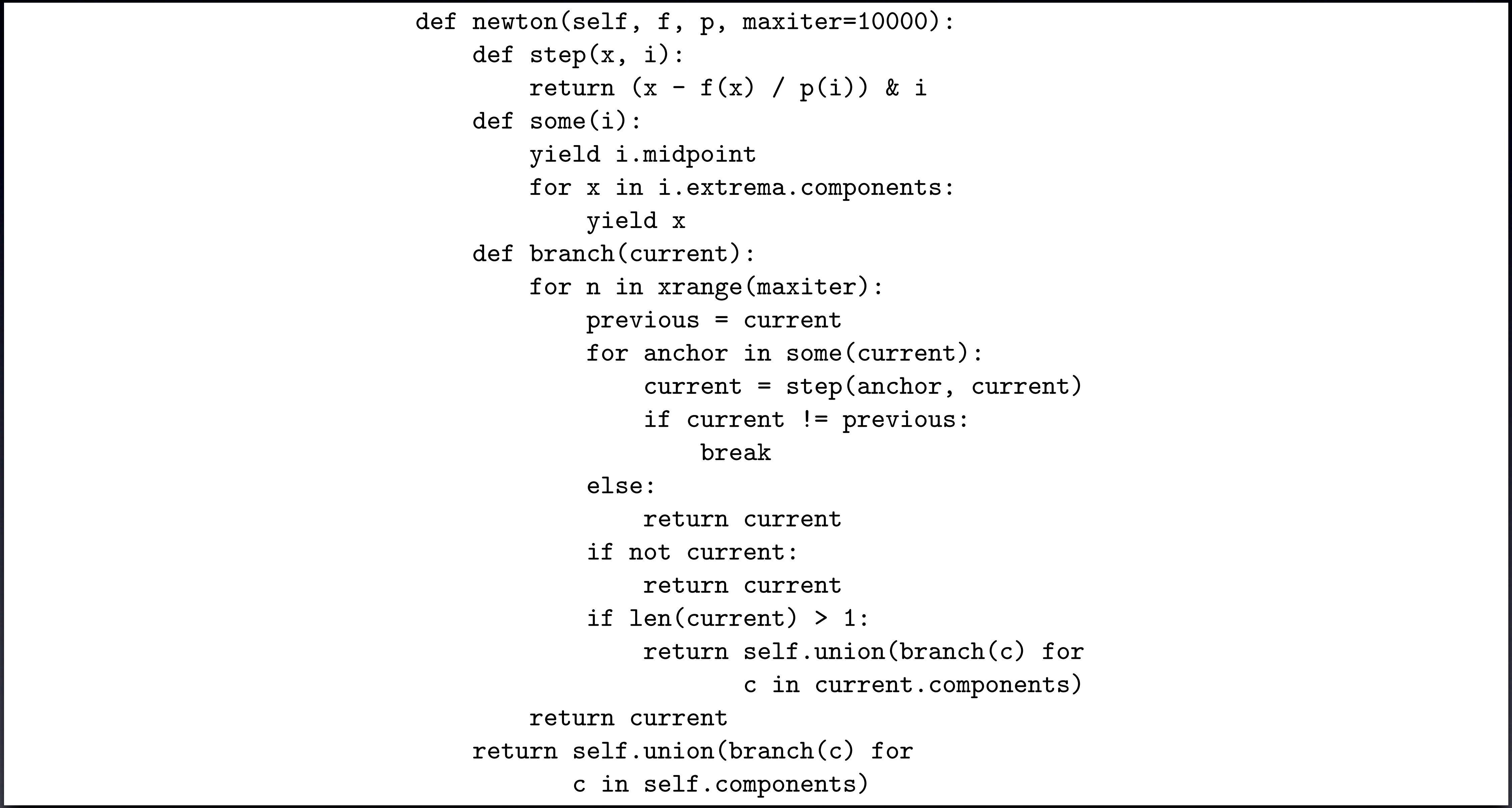
$$(x^2 - 1)(x - 2) = 0$$

```
>>> interval[-100, 100].newton(
...    lambda x: (x**2 - 1)*(x - 2),
...    lambda x: 3*x**2 - 4*x -1)
interval([-1.0], [1.0], [2.0])
```

$$\cos(\pi x/3) = 0.5$$

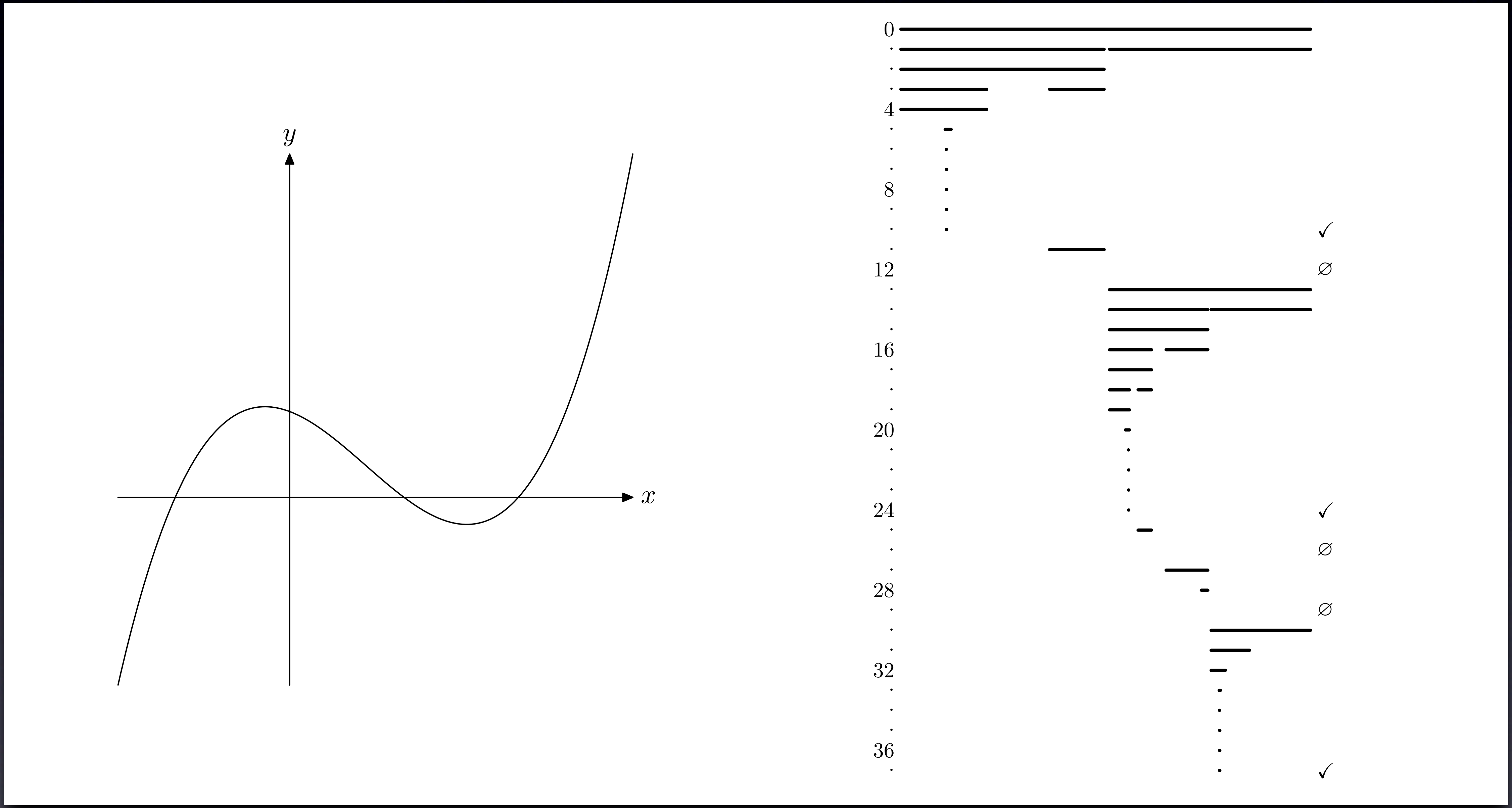
```
>>> print interval[-10, 10].newton(
...    lambda x: imath.cospi(x/3) - 0.5,
...    lambda x: -imath.pi * imath.sinpi(x/3) / 3)
interval([-7.0, -7.0], [-5.0, -5.0], [-1.0, -1.0], \
         [1.0, 1.0], [5.0, 5.0], [7.0, 7.0])
```

Newton (2)

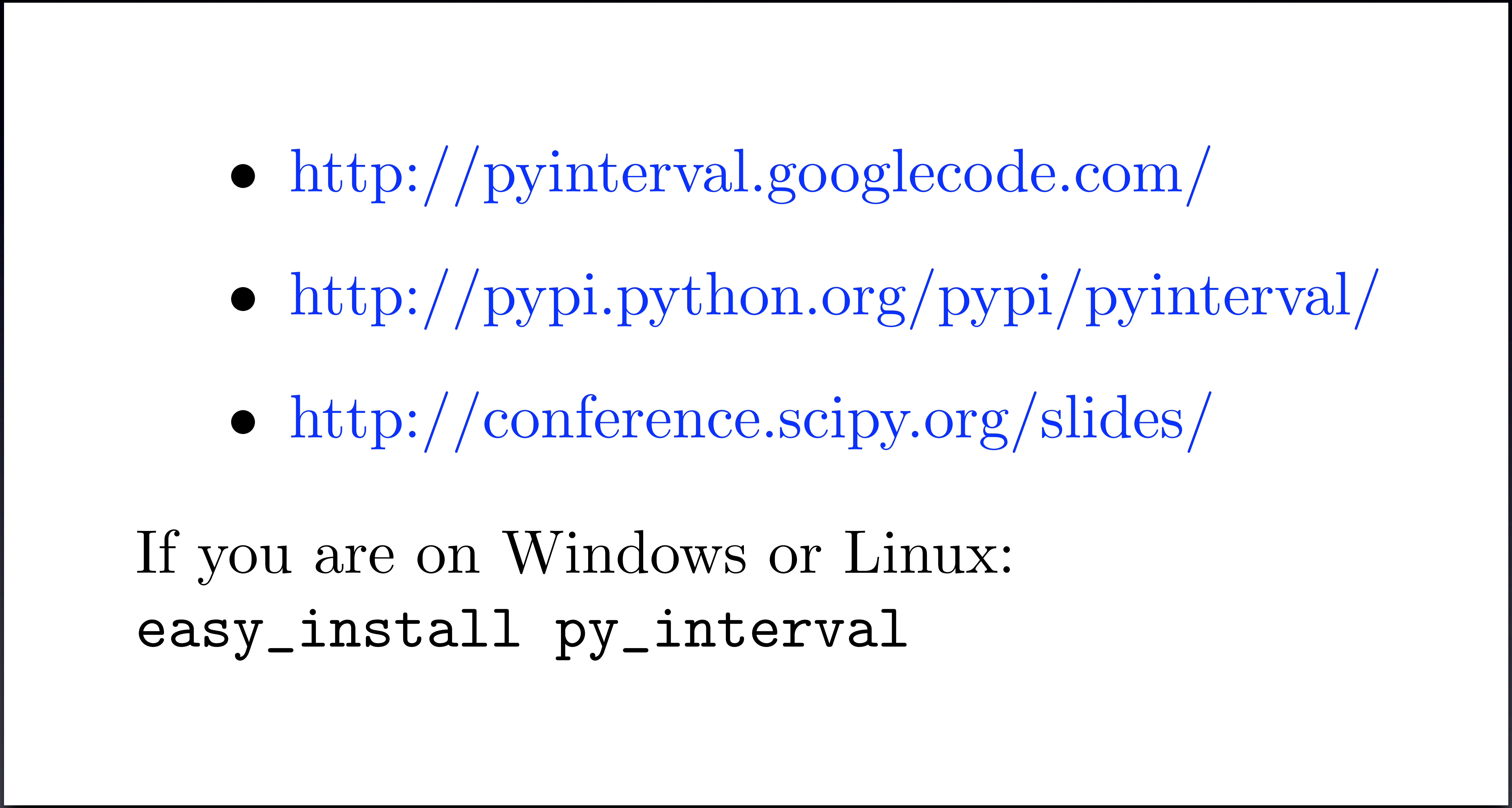


```
def newton(self, f, p, maxiter=10000):
    def step(x, i):
        return (x - f(x) / p(i)) & i
    def some(i):
        yield i.midpoint
        for x in i.extrema.components:
            yield x
    def branch(current):
        for n in xrange(maxiter):
            previous = current
            for anchor in some(current):
                current = step(anchor, current)
                if current != previous:
                    break
            else:
                return current
            if not current:
                return current
            if len(current) > 1:
                return self.union(branch(c) for
                       c in current.components)
        return current
    return self.union(branch(c) for
           c in self.components)
```

Newton (3)



Notes



- http://pyinterval.googlecode.com/
- http://pypi.python.org/pypi/pyinterval/
- http://conference.scipy.org/slides/

If you are on Windows or Linux:

```
easy_install py_interval
```