DECIMALS

Constructors and Contexts

Constructing Decimal Objects

The `Decimal` class is in the `decimal` module

```
import decimal
from decimal import Decimal
```

`Decimal(x)` `x` can be a variety of types

integers	`a = Decimal(10)`	→ `10`
other Decimal object		
strings	`a = Decimal('0.1')`	→ `0.1`
tuples	`a = Decimal((1, (3, 1, 4, 1, 5), -4))`	→ `-3.1415`

floats? yes, but not usually done

`Decimal(0.1)` → `0.1000000000000000005551`

Since `0.1` does not have an exact binary float representation
it cannot be used to create an exact Decimal representation
of itself

→ Use strings or tuples instead

Using the tuple constructor

1.23 → +123 x 10^{-2}

-1.23 → -123 x 10^{-2}

sign

digits

exponent

(s, (d1, d2, d3, ...), exp)

Example: -3.1415 → (1, (3, 1, 4, 1, 5), -4)

```
a = Decimal((1, (3, 1, 4, 1, 5), -4))
```

a → -3.1415

Context Precision and the Constructor

Context precision affects mathematical operations

Context precision does not affect the constructor

```
import decimal
from decimal import Decimal

decimal.getcontext().prec = 2     ← global (default) context now has precision set to 2

a = Decimal('0.12345')   a → 0.12345
b = Decimal('0.12345')   b → 0.12345
c = a + b                a + b = 0.2469    c → 0.25
```

Local vs Global Context

```
import decimal
from decimal import Decimal

decimal.getcontext().prec = 6


a = Decimal('0.12345')

b = Decimal('0.12345')

print(a + b)                          → 0.24690

with decimal.localcontext() as ctx:
    ctx.prec = 2
    c = a + b
    print(c)                          → 0.25

print(c)                              → 0.25
```

Code