Hyperbolic Functions

```
double cosh(double x);
double sinh(double x);
double tanh(double x);
```

cosh
sinh
tanh

The `cosh`, `sinh`, and `tanh` functions compute the hyperbolic cosine, sine, and tangent:

`cosh(0.5)` ⇒ 1.12763
`sinh(0.5)` ⇒ 0.521095
`tanh(0.5)` ⇒ 0.462117

Arguments to `cosh`, `sinh`, and `tanh` must be expressed in radians, not degrees.

Exponential and Logarithmic Functions

```
double exp(double x);
double frexp(double value, int *exp);
double ldexp(double x, int exp);
double log(double x);
double log10(double x);
double modf(double value, double *iptr);
```

exp

The `exp` function returns e raised to a power:

`exp(3.0)` ⇒ 20.0855

log
log10

`log` is the inverse of `exp`—it computes the logarithm of a number to the base e. `log10` computes the "common" (base 10) logarithm:

`log(20.0855)` ⇒ 3.0
`log10(1000)` ⇒ 3.0

Computing the logarithm to a base other than e or 10 isn't difficult. The following function, for example, computes the logarithm of `x` to the base `b`, for arbitrary `x` and `b`:

```
double log_base(double x, double b)
{
  return log(x) / log(b);
}
```

modf

The `modf` and `frexp` functions decompose a `double` value into two parts. `modf` splits its first argument into integer and fractional parts. It returns the fractional part and stores the integer part in the object pointed to by the second argument: