Finding Roots of Functions

We're looking for the point at which a function f(x) crosses the x-axis.

We know that at one of two points on the axis (x1, x2) the function is below the axis, and that at the other it's above the axis.

Uses of This

If we can find the place f(x) = (x*x) - 2 crosses the x-axis, we have found the value of sqrt(2).

In general, you can find the nth root of a value m by finding the root of the function f(x) = x^n - m.

Process (The Bisection Method)

Call the points low and high. Either f(low) < 0 and f(high) > 0, or the other way around.

We want to find z > low, z < high such that f(z) = 0.

We already know how to do this. It's an exmaple of binary search:

```
def Zero(f, lo, hi):

    mid = (lo + hi)/2
    if f(mid) == 0:
        return mid
    if f(lo) * f(mid) < 0:
        hi = mid
    else:
        low = mid
```

Note the second if condition is checking if f(lo) and f(mid) have opposite sign.

With this program (with a loop in it), it's likely that you'll never hit exactly the number you need, so you need to specify how close you want to get to the number:

```
def Zero(f, lo, hi):

    while hi - lo >= 0.001:
        mid = (lo + hi)/2
        if f(mid) == 0:
            return mid
        if f(lo) * f(mid) < 0:
            hi = mid
        else:
            low = mid
    return (lo + hi)/2
```

You can also let the user input a precision value:

```
def Zero(f, lo, hi, tolerance=0.001):

    while hi - lo >= 2*tolerance:
        mid = (lo + hi)/2
        if f(mid) == 0:
            return mid
        if f(lo) * f(mid) < 0:
            hi = mid
        else:
            low = mid
    return (lo + hi)/2
```

This function guarantees a maximum error in the answer of 'tolerance'.