CS 261 Fall 2011
The Bisection Method

Suppose that we would like to estimate the root of the function $f(x)$ between $x = a$ and $x = b$ using the bisection method. We can implement this in MATLAB (or OCTAVE) as follows. First create a file named `f.m` whose contents just define the function $f$.

```
function y=f(x)

y=x^2-2;
```

Next create a second file `bisect.m` whose contents contain the bisection algorithm applied to $f$. Be sure that the files `f.m` and `bisect.m` are in the same directory.

```
function p=bisect(a,b,tol,n)
% Output: p estimate of root
% Input: Interval [a,b], Tolerance tol, Max Interations n
% Evaluates a user written function f()

%Check intervals are opposite signs
if f(a)*f(b) >=0
    error('f(a) and f(b) do not have opposite signs')
end

% Initialize Variables
i=1;
fa=f(a) ;

while i < n
   p = a+(b-a)/2;
   fp=f(p);
   if fp==0 | (b-a)/2<tol
       break
   end
   i = i+1;
   if fa*fp > 0
       a=p;
       fa=fp;
   else
       b=p;
   end
   if i== n
       error('Max Iterations Reached, Method Failed')
   end
end
```

Running these programs with $a = 1$, $b = 2$, tolerance $1 \times 10^{-9}$, and 100 000 as the maximum number of steps, we find the following.

```
octave-3.2.3:1> format long

octave-3.2.3:2> bisect(1,2,1E-9,100000)
ans =  1.41421356052160
```

As well, we can compare our estimate to $\sqrt{2}$ to the software's internal value of $\sqrt{2}$.

```
octave-3.2.3:3> sqrt(2)
ans =  1.41421356237310
```

Note that our estimate and the software's internal value are accurate to 8 decimal places (as expected).

**Example.** Consider the function

$$f(x) = 48x(1+x)^{60} - (1+x)^{60} + 1.$$

Note that $f(0) = 0$. There is, however, another root slightly larger than 0. Using the bisection method with $a = 0.001$, $b = 0.1$, tolerance $1 \times 10^{-9}$, and 100 000 as the maximum number of steps, we find the following.

```
octave-3.2.3:4> bisect(0.0001,0.01,1E-9,100000)
ans =  0.00762860316634178
```