Exercise 1: Write a small Matlab function largest(a,b) that returns the largest of the two values. Test that your function works by computing largest(1,2), largest(0,-1) and largest(5,5).

```
function retval=largest(a,b)
    if(a>b)
        retval=a;
    else
        retval=b;
    end
endfunction
```

```
>> largest(1,2)
ans =  2
>> largest(0,-1)
ans = 0
>> largest(5,5)
ans =  5
>> diary off
```

Exercise 2: Write a small Matlab function nextprime(x) that takes a positive integer argument and returns the smallest prime number at least as large as x. Your function should use a while loop and take advantage of the isprime function in Matlab. Test that your function works by computing nextprime(5), nextprime(6), nextprime(-1) and nextprime(100).

```
function x=nextprime(x)
while(~isprime(x))
    x=x+1;
end
endfunction
```

```
>> diary on
>> nextprime(5)
ans =  5
>> nextprime(6)
ans =  7
>> nextprime(-1)
ans =  2
>> nextprime(100)
ans =  101
>> diary off
```

Exercise 3: Define a sequence of numbers by $x_1 = 1$ and $x_{k+1} = \frac{1}{2}x_k + 1$. Write a Matlab function buildseq(N) that returns an array with the first N elements of the sequence in it. For example, buildseq(2) should return [1, 0.75]. Test that your function works by computing the first four sequence elements by hand, and then verifying that your function computes them correctly. You may wish to take advantage of the Matlab command zeros.

```
function retval=buildseq(n)
retval=[];
f=@(x) .5*x+1;
x=1
for i=[1:n]
    retval=[retval x];
    x=f(x);
end
endfunction
```

```
>> diary on
>> buildseq(4)
x =  1
ans =

   1.0000   1.5000   1.7500   1.8750

>> diary off
```

Exercise Chapter 4: 2(a):

This is my bisect method

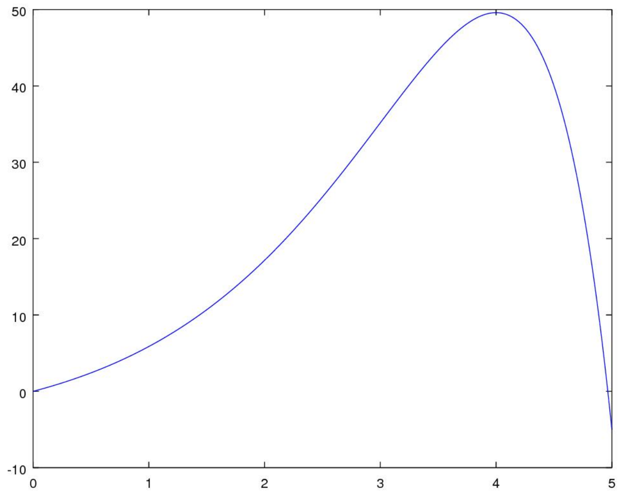
```
function retval=findzero(a,b,tol,f)
retval=(a+b)/2;
fa=f(a);
fb=f(b);
%lets make a have the negative val of f
if(sign(fa)>sign(fb))
    temp=b;
    b=a;
    a=temp;
    temp=fa;
    fa=fb;
    fb=temp;
end
err=abs(a-b)/2;
while(err>tol)
    disp([a,b]);
    err=err/2;
    temp=f(retval);
    if(sign(temp)<1)
        a=retval;
        fa=temp;
    else
        b=retval;
        fb=temp;
    end
    retval=(a+b)/2;
end
endfunction
```

The script to solve

```
f=@(x) (5-x) .* exp(x) -5;
plot([0:.01:5],f([0:.01:5]));
saveas(gcf,"q4fig.jpg");
findzero(4,5,(1e-6)/2,f)
```

And the result of the script

```
>> question_4
    5   4
    5.0000   4.5000
    5.0000   4.7500
    5.0000   4.8750
    5.0000   4.9375
    4.9688   4.9375
    4.9688   4.9531
    4.9688   4.9609
    4.9688   4.9648
    4.9668   4.9648
    4.9658   4.9648
    4.9653   4.9648
    4.9653   4.9651
    4.9652   4.9651
    4.9651   4.9651
    4.9651   4.9651
    4.9651   4.9651
    4.9651   4.9651
    4.9651   4.9651
    4.9651   4.9651
ans =  4.9651
>> diary off
```



The interval I after N iterations is $I = 2^{-N}$. Thus if we wanted a interval of $I = 10^{-12}$ we would set these as a inequality $10^{-12} \geq 2^{-N}$. Since log is monotonic increasing we can take $\log_2$ of both sides $-12\log_2 10 \geq -N$. so we finaly get $12\log_2 10 \geq N$ and the smallest natural with this property is $N = 40$.

Exercise Chapter 4: 2(b):

My newton method

```
function guess=newton(guess,f,df,tol_y)
step=@(x) x-f(x)/df(x);
err_y=@(x) abs(f(x));
%abselute max number of steps, sometimes this method does not converge
N=10000;
n=0;
px=[];
py=[];
while(N>0 && err_y(guess)>tol_y)
    N=N-1;
    guess=step(guess);
    disp([guess, f(guess)]);
    px=[px,n];
    py=[py,f(step(guess))/f(guess)];
    n=n+1;
end
plot(px,abs(py),'x-');
endfunction
```

FINISH LATER

Exercise Chapter 4: 18: