

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)`.

```
1 function retval=largest(a,b)
2     if(a>b)
3         retval=a;
4     else
5         retval=b;
6     end
7 endfunction
```

```
1 >> largest(1,2)
2 ans = 2
3 >> largest(0,-1)
4 ans = 0
5 >> largest(5,5)
6 ans = 5
7 >> 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)`.

```
1 function x=nextprime(x)
2 while (~isprime(x))
3     x=x+1;
4 end
5 endfunction
```

```
1 >> diary on
2 >> nextprime(5)
3 ans = 5
4 >> nextprime(6)
5 ans = 7
6 >> nextprime(-1)
7 ans = 2
8 >> nextprime(100)
9 ans = 101
10 >> 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, 1.5]`. 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`.

```
1 function retval=buildseq(n)
2 retval=[];
3 f=@(x) .5*x+1;
4 x=1
5 for i=[1:n]
6     retval=[retval x];
7     x=f(x);
8 end
9 endfunction
```

```
1 >> diary on
2 >> buildseq(4)
3 x = 1
4 ans =
5
6     1.0000     1.5000     1.7500     1.8750
7
8 >> diary off
```

Exercise Chapter 4: 2(a):

This is my bisection method

```
1 function retval=findzero(a,b,tol,f)
2 retval=(a+b)/2;
3 fa=f(a);
4 fb=f(b);
5 %lets make a have the negative val of f
6 if(sign(fa)>sign(fb))
7     temp=b;
8     b=a;
9     a=temp;
10    temp=fa;
11    fa=fb;
12    fb=temp;
13 end
14 err=abs(a-b)/2;
15 while(err>tol)
16     disp([a,b]);
17     err=err/2;
18     temp=f(retval);
19     if(sign(temp)<1)
20         a(retval);
21         fa=temp;
22     else
23         b(retval);
24         fb=temp;
25     end
26     retval=(a+b)/2;
27 end
28 endfunction
```

The script to solve

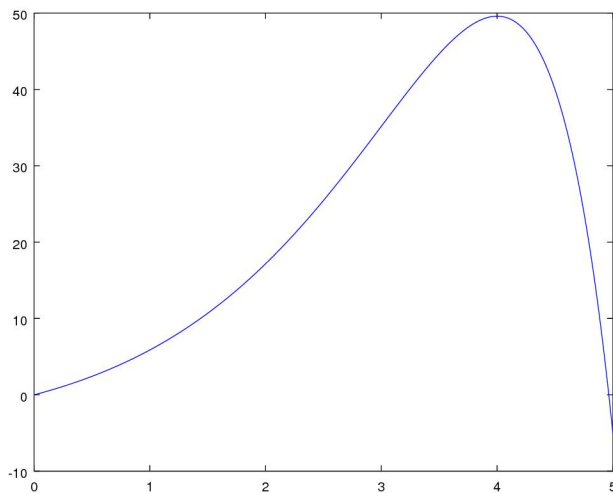
```
1 f=@(x) (5-x) .* exp(x) -5;
2 plot([0:.01:5],f([0:.01:5]));
3 saveas(gcf,"q4fig.jpg");
4 findzero(4,5,(1e-6)/2,f)
```

And the result of the script

```

1 >> question_4
2     5     4
3     5.0000    4.5000
4     5.0000    4.7500
5     5.0000    4.8750
6     5.0000    4.9375
7     4.9688    4.9375
8     4.9688    4.9531
9     4.9688    4.9609
10    4.9688    4.9648
11    4.9668    4.9648
12    4.9658    4.9648
13    4.9653    4.9648
14    4.9653    4.9651
15    4.9652    4.9651
16    4.9651    4.9651
17    4.9651    4.9651
18    4.9651    4.9651
19    4.9651    4.9651
20    4.9651    4.9651
21    4.9651    4.9651
22 ans = 4.9651
23 >> 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 final 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

```
1 function guess=newton(guess,f,df,tol_y)
2 step=@(x) x-f(x)/df(x);
3 err_y=@(x) abs(f(x));
4 %absolute max number of steps, sometimes this method does not converge
5 N=10000;
6 while (N>0 && err_y(guess)>tol_y)
7     N=N-1;
8     guess=step(guess);
9     disp([guess, f(guess)]);
10 end
11 endfunction
```

FINISH LATER NOT DUE TILL NEXT WEEK

Exercise Chapter 4: 18:

let's begin by calculating the derivatives of $f(x) = e^{1-x^2}$.

$$f(x) = e^{1-x^2}$$

$$f'(x) = -2xe^{1-x^2}$$

$$f''(x) = (4x^2 - 2)e^{1-x^2}$$

$$f'''(x) = (12x - 8x^3)e^{1-x^2}$$

Now that we have these it is trivial to crate a Taylor expansion.

```

1  x_0=1;
2  f=@(x) exp(1-x.^2);
3  fp=@(x) (-2*x).*exp(1-x.^2);
4  fpp=@(x) (4*x.^2-2).*exp(1-x.^2);
5  fppp=@(x) (12*x-8*x.^3).*exp(1-x.^2);
6
7  x=[-4:.01:4];
8
9
10
11 p_0=@(x) f(x_0)+x-x;
12 p_1=@(x) p_0(x)+fp(x_0).*(x-x_0);
13 p_2=@(x) p_1(x)+1/2*fpp(x_0).*(x-x_0).^2;
14 p_3=@(x) p_2(x)+1/6*fppp(x_0).*(x-x_0).^3;
15 y=[p_0(x); p_1(x); p_2(x); p_3(x)];
16
17
18 for i=1:4
19     subplot(2,2,i);
20     plot(x,f(x),'k--',x,y(i,:), 'k-');
21     axis([-4,4,-.1,3]);
22     title(strcat("Plot of P_",num2str(i-1),"(x) and f(x)"));
23 end
24
25 saveas(gcf,"q5fig.jpg");

```

