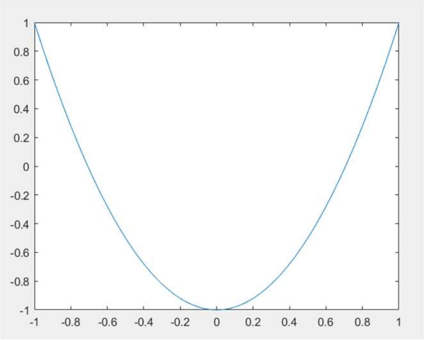
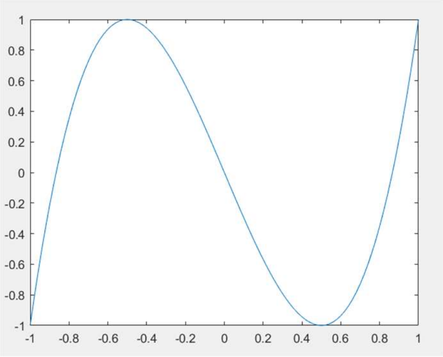
1)

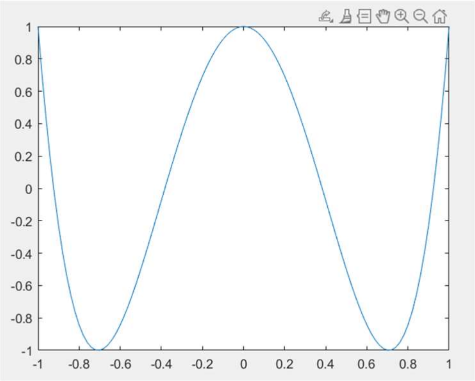
```
function y = polyout(c, x)
% Function that grabs a row coefficients c and a vector of values x of any polynomial
% and outputs a vector of values
% y = polyout(c, x)
% plot(x, polyout(c, x))

%length of the coefficients
cl = length(c);

y = zeros(size(x)); %vectors of 0 the size of x

for i = 1: cl
    y = x .* y + c(i);
end
end

c1 = [2 0 -1];
x1 = linspace(-1,1,101);
plot(x1,polyout(c1, x1))
```



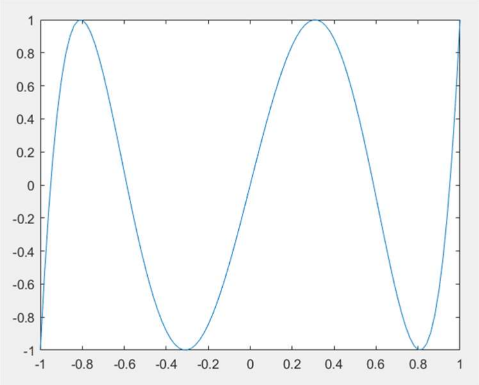
```
c2 = [4 0 -3 0];
x2 = linspace(-1,1,101);
plot(x2,polyout(c2, x2))
```



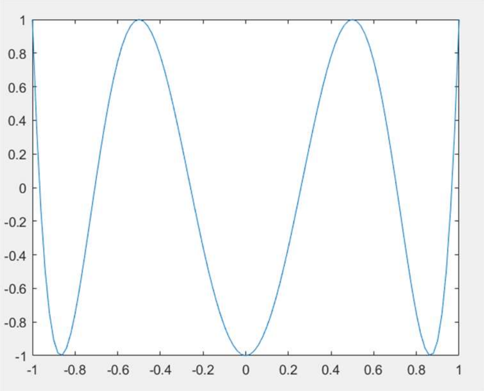
```
c3 = [8 0 -8 0 1];
x3 = linspace(-1,1,101);
plot(x3,polyout(c3, x3))
```



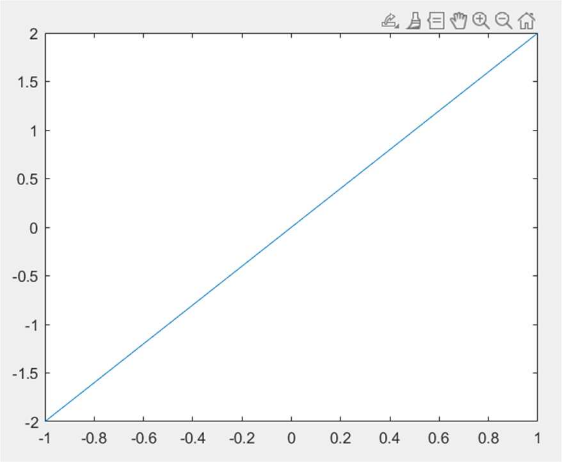
```
c4 = [16 0 -20 0 5 0];
x4 = linspace(-1,1,101);
plot(x4,polyout(c4, x4))
```



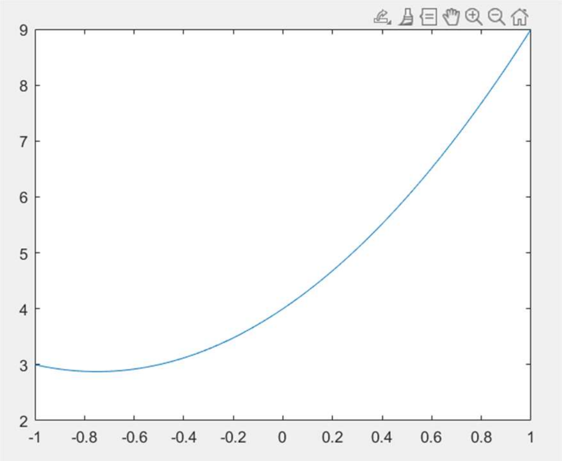
```
c5 = [32 0 -48 0 18 0 -1];
x5 = linspace(-1,1,101);
plot(x5,polyout(c5, x5))
```



```
INDEPENDENT TESTING

c6 = [2 0];
x6 = linspace(-1,1,101);
plot(x6,polyout(c6, x6))
```



```
c7 = [2 3 4];
x7 = linspace(-1,1,101);
plot(x7,polyout(c7, x7))
```

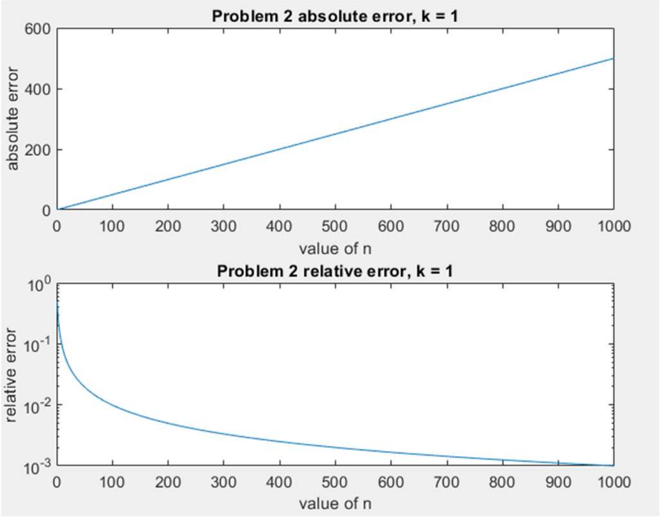


```
diary off
```

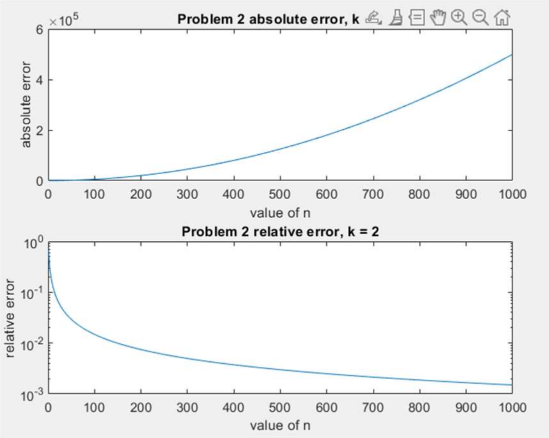
```
------------------------------------------------------------------------------------
```

2)

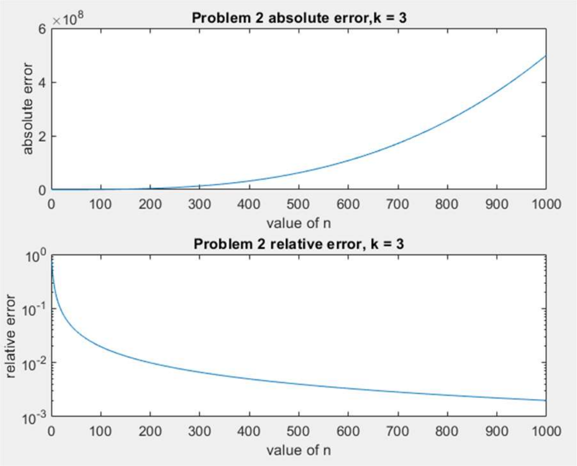
```
function  [abs_err , rel_err] = errors(k ,n)
%[abs_err, rel_err] = errors(k, n)
%Finds the approx and uses the approx to graph absolute and relative error
%plot(1:n, x)
%plot(1:n y)

v = 1 : n; % vector to n
a = v .^ (k + 1) / (k + 1);% aprox

vpk = cumsum(v .^ k); %vector to the power of k cumulative sum

abs_err = abs(vpk - a); %absolute error
rel_err = abs_err ./ vpk; %relative error
end

N = 1:1000;
[abs_err1, rel_err1] = errors(1, 1000);

subplot(2,1,1)
plot(N, abs_err1)
title('Problem 2 absolute error, k = 1'); xlabel('value of n'); ylabel('absolute
error'); hold off;

subplot(2,1,2)
semilogy(N, rel_err1)
title('Problem 2 relative error, k = 1'); xlabel('value of n'); ylabel('relative
error'); hold off;
```



```
[abs_err2, rel_err2] = errors(2, 1000);

subplot(2,1,1)
plot(N, abs_err2)
title('Problem 2 absolute error, k = 2'); xlabel('value of n'); ylabel('absolute
error'); hold off;

subplot(2,1,2)
semilogy(N, rel_err2)
title('Problem 2 relative error, k = 2'); xlabel('value of n'); ylabel('relative
error'); hold off;
```

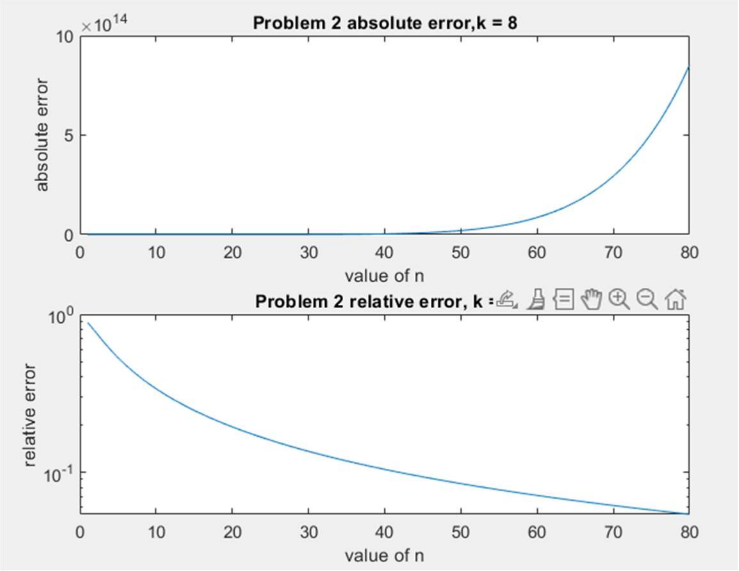


```
[abs_err3, rel_err3] = errors(3, 1000);

subplot(2,1,1)
plot(N, abs_err3)
title('Problem 2 absolute error,k = 3'); xlabel('value of n'); ylabel('absolute
error'); hold off;

subplot(2,1,2)
semilogy(N, rel_err3)
title('Problem 2 relative error, k = 3'); xlabel('value of n'); ylabel('relative
error'); hold off;
```



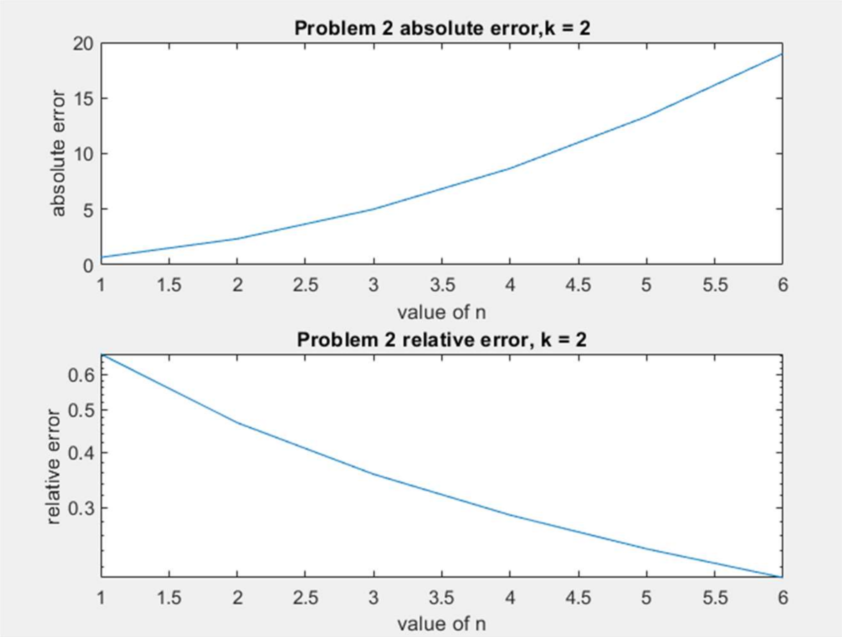
INDEPENDNT TESTING

```
N = 1:80;
[abs_err, rel_err] = errors(8, 80);

subplot(2,1,1)
plot(N, abs_err)
title('Problem 2 absolute error,k = 8'); xlabel('value of n'); ylabel('absolute
error'); hold off;

subplot(2,1,2)
semilogy(N, rel_err)
title('Problem 2 relative error, k = 8'); xlabel('value of n'); ylabel('relative
error'); hold off;
```



```
N = 1:6;
[abs_err, rel_err] = errors(2, 6);

subplot(2,1,1)
plot(N, abs_err)
title('Problem 2 absolute error,k = 2'); xlabel('value of n'); ylabel('absolute
error'); hold off;

subplot(2,1,2)
semilogy(N, rel_err)
title('Problem 2 relative error, k = 2'); xlabel('value of n'); ylabel('relative
error'); hold off;

v =

     1     2     3     4     5     6
a =

    0.3333    2.6667    9.0000   21.3333   41.6667   72.0000
vpk =

     1     5    14    30    55    91
```

```
abs_err =

    0.6667    2.3333    5.0000    8.6667   13.3333   19.0000

rel_err =

    0.6667    0.4667    0.3571    0.2889    0.2424    0.2088
```



--------------------------------------------------------------------------------------------------------------------------------------------

3)

Part 1:

```
function [w1] = problem3_part1(A,v,c)
%[w1] = problem_3_method1(A,v,c)
%   first method successively computes the matrix powers
k = length(c);
B = eye(1000);
term = c(1)*B; %does the first term of c1v

  % Goes through the length of c, then successively computes the matrix powers and
%outputs term
  for i = 2 : k
    B = A * B;
    term = term + c(i) *B;
  end

  w1 = term * v; %computes w = Bv
end
```

```
A = [ 1 2 3; 4 5 6; 7 8 9];
V = [ 1; 2; 3];
C = [1,2];
z=problem3_part1(A,v,c)
Z = 29
    66
    103
```

```
A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:4];
```

```
tic, z = problem3_part1(A,v,c); toc
Elapsed time is 0.243485 seconds.
```

```
A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:8];
```

```
tic, z = problem3_part1(A,v,c); toc
Elapsed time is 0.318256 seconds.
```

```
A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:6];
```

```
tic, z = problem3_part1(A,v,c); toc
Elapsed time is 0.219123 seconds.
```

```
A =round(10*rand(500)-5);
v = [1:500]';
c = [1:4];
```

```
tic, z = problem3_part1(A,v,c); toc
Elapsed time is 0.022728 seconds.
```

part 2:

```
function [term] = problem3_part2(A, v, c)
%[term] = problem_3_part2(A,v,c)
% Second method reads from left to right by successively computing Av and
% computes for a column vector

k = length(c);
term = c(1)* v ; %calculates first case cv1
Av = A * v; %computes A by v' will be our w
% Takes
for i = 2 : k
    term = term + c(i) * Av;
    Av = A * Av;%
end
end
```

```
A = [ 1 2 3; 4 5 6; 7 8 9];
V = [ 1; 2; 3];
C = [1,2];
z=problem3_part2(A,v,c)
Z = 29
    66
    103

A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:4];


tic, z = problem3_part2(A,v,c); toc
Elapsed time is 0.004989 seconds.


A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:8];

tic, z = problem3_part2(A,v,c); toc
Elapsed time is 0.010371 seconds.

A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:6];

tic, z = problem3_part2(A,v,c); toc
Elapsed time is 0.004167 seconds.

A =round(10*rand(500)-5);
v = [1:500]';
c = [1:4];

tic, z = problem3_part2(A,v,c); toc
Elapsed time is 0.002637 seconds
```

part3:

```
function [term] = problem3_part3(A, v, c)
%[term] = problem_3_part3(A,v,c)
% Third method reads from right to left by successively computing term A and
% computes for a column vector

k = length(c);
term = c(end) * v;%calculates first case cv1

for i = k - 1 :-1 :1
    term = A * term + c(i) * v;
end
end

A = [ 1 2 3; 4 5 6; 7 8 9];
V = [ 1; 2; 3];
C = [1,2];
z=problem3_part3(A,v,c)
Z = 29
    66
    103


A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:4];

tic, z = problem3_part3(A,v,c); toc
Elapsed time is 0.005740 seconds.

A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:8];

tic, z = problem3_part3(A,v,c); toc
Elapsed time is 0.005431 seconds.

A =round(10*rand(1000)-5);
v = [ 1:1000]';
c = [1:6];

tic, z = problem3_part3(A,v,c); toc
Elapsed time is 0.004600 seconds.

A =round(10*rand(500)-5);
v = [1:500]';
c = [1:4];

tic, z = problem3_part3(A,v,c); toc
Elapsed time is 0.000935 seconds.
```

*Conclusion for question 3, Doing matrix by matrix multiplication is very expensive and is generally slower and the other 2 methods were much faster getting a result.*