Sachin Srivastava  
Numerical Analysis  
Assignment 1

1)

i)

function [ result ,error] = fun( z,a,n )  
-> Define a function which takes 3 arguments.

syms x real;  
-> Define a symbol x.

f=sin(x);  
-> Define a function sin(x)

sum=subs(f,'x',a);  
-> Initialize sum as f(a)

prod=1;  
-> Initialize prod as 1.

for j=1:n  
-> for loop with j, 1 to n.

prod=prod\*(z-a)/j;  
 sum=sum+prod\*subs(diff(f,x,j),'x',a);  
-> Calculate Taylors series

end

result=double(sum);

error=double(abs(result-subs(f,'x',z)));  
-> Calculate Error.

end…………………………………………………………………………………………………………………………………………………………………

z=input('z=');  
-> input z value from user

a=input('a=');  
-> input a value from user

for n=1:10

[ result ,error] = fun( z,a,n );  
-> Call function with different n values.

v(n)=result;e(n)=error;  
-> Store result and error

end  
-> End for loop.

x=1:n;

disp( ' n n(1) error')

disp([x' v' e'])  
-> Print the value and error.

fid=fopen('expsin.txt','w');

fprintf(fid,'%s\n',' n Pn(1) error');

fprintf(fid,'%2u %14.10f %14.10f\n',[x;v;e]);  
-> Print in a file.

fclose(fid);

ii)

n f n(1)

[ 1, 1.0, 0.84147098480789650487565722869476]

[ 2, 1.0, 0.84147098480789650487565722869476]

[ 3, 0.83333333333333333333333333333333, 0.84147098480789650487565722869476]

[ 4, 0.83333333333333333333333333333333, 0.84147098480789650487565722869476]

[ 5, 0.84166666666666666666666666666667, 0.84147098480789650487565722869476]

[ 6, 0.84166666666666666666666666666667, 0.84147098480789650487565722869476]

[ 7, 0.84146825396825396825396825396825, 0.84147098480789650487565722869476]

[ 8, 0.84146825396825396825396825396825, 0.84147098480789650487565722869476]

[ 9, 0.84147100970017640886311482972815, 0.84147098480789650487565722869476]

[ 10, 0.84147100970017640886311482972815, 0.84147098480789650487565722869476]

iii)  
For n=10, max |Rn(x)|using above table vs error using Rn(x) formula is  
0.000000024892279903987457601033383980393, -0.000000025052108385441718775052108385442  
Which is very close to each other.

iv)  
[ 1, 0.15852901519210349512434277130524]

[ 2, 0.15852901519210349512434277130524]

[ 3, 0.008137651474563134534889741189545]

[ 4, 0.008137651474563134534889741189545]

[ 5, 0.00019568185877016919249626880628057]

[ 6, 0.00019568185877016919249626880628057]

[ 7, 0.000002730839642528515298636193620041]

[ 8, 0.000002730839642528515298636193620041]

[ 9, 0.000000024892279903987457601033383980393]

[ 10, 0.000000024892279903987457601033383980393]

v)  
For z=1, a=0,   
|𝑃𝑛 (𝑧) − 𝑃𝑚(𝑧)| is  
m=6, n=1 => 0.15833333333333333333333333333333

m=6, n=1 => 0.15853174603174603174603174603175

m=6, n=1 => 0.0081349206349206349206349206349206

m=6, n=1 => 0.0081376763668430755297814963948137

m=6, n=1 => 0.00019565696649025780355183693851965

2.  
The largest integer is 2,147,483,647. This is equal to 2^31-1, as the first bit is for sign and the rest 31 are for the number. The smallest integer is 2,147,483,648 as its represented in twos compliment form. The range for floating point numbers is -3.40282e+38 to -1.17549e-38 and 1.17549e-38 to 3.40282e+38. Which is a range between (2- 2^-23) \*2^127 and 2^-126.

3  
i) The decimal number 4/3 cannot be exactly converted into a binary fraction. So the above calculation does not result in zero. The answer comes as 2^-52 which is a precision value.

ii)This is because 0.1 is not exactly represented in binary number.

iii) This is because of the order of operation.

iv) subtractions are performed with nearly equal operands, sometimes cancellation can occur unexpectedly. loss of precision

4.  
sin(10^-7) = 1.0000e-07  
sqrt(1-(cos(10^-7))^2) = 9.9960e-08

Sin(10^-8) = 1.0000e-08  
sqrt(1-(cos(10^-8))^2) = 0

This is because there is a loss of precision when we are subtracting the numbers.

5.  
If we substitute 10^(-12), we get 1.0001. For numbers less than this, the value is even lesser. Using the taylors polynomial we should actually get 1. This is because when we expand taylor polynomial for e^x. we get 1+x/1! + x^2/2! + x^3/3! + … If we subtract by 1 and divide by x we get, 1+x/2! + x^2/3!+…