x=input('Is equation is a polynomial (Y/N) \n','s');

clc

if ~strcmp(x,'Y')

fprintf('(1) Bisection \n(2) False-position \n(3) Fixed-Point \n(4) Newton-Raphson \n(5) Secant \n');

y=input('');

clc

if y==1

i=0;

flag=0;

f\_x = input('Give an equation in x:\n','s');

d=input('Enter two starting points using [a b]\n');

fprintf('Stopping criteria:\n(i) Convergence criterion for relative approximate errors in successive iterations. The approximate relative error may be for the root for some methods, interval for some methods and factor values for some methods. \n(ii) Convergence criteria for the function value, i.e., how close f(x) is to zero. \n(iii) Maximum iteration number (an integer)\n');

fprintf('Enter maximum relative approximate error and number of iterations line by line\n');

tol=input('');

n=input('');

b=d(2);

a=d(1);

x1=[];

y1=[];

%disp(feval(inline(f\_x,'x'),a));

%disp(feval(inline(f\_x,'x'),b));

while abs(b-a)>=tol/100

c=(a+b)/2;

i=i+1;

x1=[x1,i];

y1=[y1,abs(b-a)];

fc=feval(inline(f\_x,'x'),c);

%disp(fc);

%disp(abs(b-a));

if feval(inline(f\_x,'x'),a)\*fc<0

b=c;

fb=fc;

elseif feval(inline(f\_x,'x'),b)\*fc<0

a=c;

fa=fc;

else

flag=2;

break

end

if i==n

flag=3;

break;

end

if abs(b-a)>tol

flag=1;

end

end

disp(c);

disp(flag);

x = -20:pi/100:20;

y = feval(inline(f\_x,'x'),x);

plot(x,y);

figure;plot(x1,y1);

end

if y==2

i=0;

flag=0;

f\_x = input('Give an equation in x:\n','s');

d=input('Enter two starting points using [a b]\n');

fprintf('Stopping criteria:\n(i) Convergence criterion for relative approximate errors in successive iterations. The approximate relative error may be for the root for some methods, interval for some methods and factor values for some methods. \n(ii) Convergence criteria for the function value, i.e., how close f(x) is to zero. \n(iii) Maximum iteration number (an integer)\n');

fprintf('Enter maximum relative approximate error and number of iterations line by line\n');

tol=input('');

n=input('');

b=d(2);

a=d(1);

x1=[];

y1=[];

while abs(b-a)>=tol/100

i=i+1;

c=a-((b-a)/(feval(inline(f\_x,'x'),b)-feval(inline(f\_x,'x'),a)))\*feval(inline(f\_x,'x'),a);

x1=[x1,i];

y1=[y1,abs(b-a)];

fc=feval(inline(f\_x,'x'),c);

%disp(feval(inline(f\_x,'x'),a)\*fc);

if feval(inline(f\_x,'x'),a)\*fc<0

b=c;

fb=fc;

elseif feval(inline(f\_x,'x'),b)\*fc<0

a=c;

fa=fc;

else

break

end

if i==n

flag=3;

break

end

if abs(b-a)>tol

flag=1;

end

end

disp(c);

disp(flag);

x = -20:pi/100:20;

y = feval(inline(f\_x,'x'),x);

plot(x,y);

figure;plot(x1,y1);

end

if y==3

i=0;

flag=0;

f\_x = input('Give an equation in x:\n','s');

g\_x=input('Enter g(x) such that f(x) can be rearranged as x=g(x) and one starting point.\n','s');

d=input('');

fprintf('Stopping criteria:\n(i) Convergence criterion for relative approximate errors in successive iterations. The approximate relative error may be for the root for some methods, interval for some methods and factor values for some methods. \n(ii) Convergence criteria for the function value, i.e., how close f(x) is to zero. \n(iii) Maximum iteration number (an integer)\n');

fprintf('Enter maximum relative approximate error and number of iterations line by line\n');

tol=input('');

n=input('');

a=d;

x1=[];

y1=[];

b=feval(inline(g\_x,'x'),a);

%disp(feval(inline(f\_x,'x'),a));

%disp(b);

while abs((b-a)/a)>tol/100

i=i+1;

a=b;

b=feval(inline(g\_x,'x'),a);

i=i+1;

fc=feval(inline(f\_x,'x'),b);

x1=[x1,i];

y1=[y1,abs((b-a)/a)];

%disp(fc);

%disp(abs((b-a)/b));

if feval(inline(f\_x,'x'),a)==0

flag=2;

break

end

if i==n

flag=3;

break

end

if abs((b-a)/a)>tol

flag=1;

end

end

disp(a);

disp(flag);

x = -20:pi/100:20;

y = feval(inline(f\_x,'x'),x);

plot(x,y);

figure;plot(x1,y1);

end

if y==4

i=0;

flag=0;

f\_x = input('Give an equation in x:\n','s');

g\_x=input('Enter f''(x) and one starting point.\n','s');

d=input('');

fprintf('Stopping criteria:\n(i) Convergence criterion for relative approximate errors in successive iterations. The approximate relative error may be for the root for some methods, interval for some methods and factor values for some methods. \n(ii) Convergence criteria for the function value, i.e., how close f(x) is to zero. \n(iii) Maximum iteration number (an integer)\n');

fprintf('Enter maximum relative approximate error and number of iterations line by line\n');

tol=input('');

n=input('');

a=d;

x1=[];

y1=[];

b=a-(feval(inline(f\_x,'x'),a)/feval(inline(g\_x,'x'),a));

%disp(feval(inline(f\_x,'x'),a));

%disp(b);

while abs((b-a)/a)>tol/100

i=i+1;

a=b;

b=a-(feval(inline(f\_x,'x'),a)/feval(inline(g\_x,'x'),a));

i=i+1;

fc=feval(inline(f\_x,'x'),b);

x1=[x1,i];

y1=[y1,abs((b-a)/a)];

%disp(fc);

%disp(abs((b-a)/b));

if feval(inline(f\_x,'x'),a)==0

flag=2;

break

end

if i==n

flag=3;

break

end

if abs((b-a)/b)>tol

flag=1;

end

end

disp(a);

disp(flag);

x = -20:pi/100:20;

y = feval(inline(f\_x,'x'),x);

plot(x,y);

figure;plot(x1,y1);

end

if y==5

i=0;

flag=0;

f\_x = input('Give an equation in x:\n','s');

d=input('Enter two starting points using [a b]\n');

fprintf('Stopping criteria:\n(i) Convergence criterion for relative approximate errors in successive iterations. The approximate relative error may be for the root for some methods, interval for some methods and factor values for some methods. \n(ii) Convergence criteria for the function value, i.e., how close f(x) is to zero. \n(iii) Maximum iteration number (an integer)\n');

fprintf('Enter maximum relative approximate error and number of iterations line by line\n');

tol=input('');

n=input('');

x1=[];

y1=[];

x\_prev=d(1);

x\_current=d(2);

x\_next=x\_current-((x\_current-x\_prev)/(feval(inline(f\_x,'x'),x\_current)-feval(inline(f\_x,'x'),x\_prev)))\*feval(inline(f\_x,'x'),x\_current);

while abs((x\_next-x\_current)/x\_current)>=tol/100

i=i+1;

x1=[x1,i];

y1=[y1,abs((x\_next-x\_current)/x\_current)];

x\_prev=x\_current;

x\_current=x\_next;

x\_next=x\_current-((x\_current-x\_prev)/(feval(inline(f\_x,'x'),x\_current)-feval(inline(f\_x,'x'),x\_prev)))\*feval(inline(f\_x,'x'),x\_current);

fc=feval(inline(f\_x,'x'),x\_next);

%disp(feval(inline(f\_x,'x'),a)\*fc);

if feval(inline(f\_x,'x'),a)\*fc<0

b=c;

fb=fc;

elseif feval(inline(f\_x,'x'),b)\*fc<0

a=c;

fa=fc;

else

break

end

if i==n

flag=3;

break

end

if abs((x\_next-x\_current)/x\_current)>=tol

flag=1;

end

end

disp(x\_next);

disp(flag);

x = -20:pi/100:20;

y = feval(inline(f\_x,'x'),x);

plot(x,y);

figure;plot(x1,y1);

end

else

s=input('Enter the function ','s'); %% taking input from the user

s1='@(x)';

c=strcat(s1,s);

f=str2func(c);

%n=input('Enter the degree of the polynomial\n');

%a=input('Enter the coefficients as [a0 a1 a2 .... an]\n');

fprintf('(1) Muller \n(2) Bairstow \n');

y=input('');

if y==1

disp('Enter starting value x1');

x\_ppprev=input('');

disp('Enter starting value x2');

x\_pprev=input('');

disp('Enter starting value x3');

x\_prev=input('');

disp('Enter max no of iterations ');

max\_no\_interation=input('');

disp('Enter max\_relative\_error ');

max\_relative\_error=input('');

x1=[];

y1=[];

w=2;

rel\_error=100\*ones(max\_no\_interation,1);

flag=0;

while(abs(rel\_error(i-1))>max\_relative\_error & i-2< max\_no\_interation)

a=((f(x\_prev)-f(x\_pprev))/(x\_prev-x\_pprev)- (f(x\_pprev)-f(x\_ppprev))/(x\_pprev-x\_ppprev))/(x\_prev-x\_ppprev);

b=(f(x\_prev)-f(x\_pprev))/(x\_prev-x\_pprev)+a\*(x\_prev-x\_pprev);

c=f(x\_prev);

i=i+1;

x1=[x1,i];

y1=[y1,abs(rel\_error(i-1))];

if(b<0)

delta\_x=-2\*c/(b-sqrt(b^2-4\*a\*c));

else

delta\_x=-2\*c/(b+sqrt(b^2-4\*a\*c));

end

x\_next=x\_prev+delta\_x;

if(i-3>0)

rel\_error(i-1)=((x\_next-x\_prev)\*100)/x\_next;

end

if(abs(rel\_error(i-1))>max\_relative\_error)

flag=1;

end

if(w==max\_no\_interation)

flag=3;

end

x\_ppprev=x\_pprev;

x\_pprev=x\_prev;

x\_prev=x\_next;

end

plot(x1,y1);

disp(x\_prev);

disp(flag);

end

if y==2;

coff=input('Enter the coeff in Increasing Order in []:');

r=input('Enter s :');

s=input('Enter r :');

maxit=input('Enter Maximum no.of Iterations :');

errmax=input('Enter the Maximum Relative approximate error in(%) :');

degree=length(coff)-1;

scoff=coff;

sdegree=degree;

arrc=[];

d=degree;

arrb=[];

flag=0;

count=0;

yr=[];

ys=[];

iee=0;

x1=[];

while(degree>0)

iee=iee+1;

x1=[x1,iee];

for j=1:maxit

d=degree;

arrb(d+1)=coff(d+1);

arrb(d)=coff(d)+r\*arrb(d+1);

d=d-2;

while(d>=0)

arrb(d+1)=coff(d+1) + r\*arrb(d+2) + s\*arrb(d+3);

d=d-1;

end

d=degree;

arrc(d+1)=arrb(d+1);

arrc(d)=arrb(d)+ r\*arrc(d+1);

d=d-2;

while(d>=0)

arrc(d+1)=arrb(d+1) + r\*arrc(d+2) + s\*arrc(d+3);

d=d-1;

end

ds = ( arrb(1)\*arrc(3) - arrb(2)\*arrc(2) )/( arrc(4)\*arrc(2) - arrc(3)\*arrc(3) );

dr = ( arrb(1)\*arrc(4) - arrb(2)\*arrc(3) )/( arrc(3)\*arrc(3) - arrc(2)\*arrc(4) );

r=r+dr;

s=s+ds;

err\_r = abs(dr/r)\*100 ;

err\_s = abs(ds/s)\*100 ;

yr=[yr,abs(dr/r)\*100];

ys=[ys,abs(ds/s)\*100];

if ( errmax > err\_s || errmax > err\_r )

root1 = ( r + sqrt(r\*r + 4\*s) )/2;

root2 = ( r - sqrt(r\*r + 4\*s) )/2;

fprintf('Solution is ');

if(count==0)

fprintf(' %f %f ',root1,root2);

end

count=1;

flag=1;

break;

end

end

if flag==0

break;

end

for i=1:degree-1

coff(i)=arrb(i+2);

end

degree=degree-2;

if degree == 2

root1 = ( -(arrb(4)) + sqrt(arrb(4)\*arrb(4) - 4\*arrb(3)\*arrb(5)) )/(2\*arrb(5));

root2 = ( -(arrb(4)) - sqrt( arrb(4)\*arrb(4) - 4\*arrb(3)\*arrb(5) ))/(2\*arrb(5));

fprintf('%f %f\n ',root1,root2);

break;

elseif degree == 1

root = -arrb(3)/arrb(4);

fprintf('%f\n ',root);

break;

end

end

f=zeros(1,101);

for k=-50:50

for i=1:sdegree+1

f(k+51)=f(k+51)+scoff(i)\*((k)^(i-1)); %x^4-7.4\*x^3+20.44\*x^2-24.184\*x+9.6448 [9.6448 -24.184 20.44 -7.4 1]

end

end

disp(iee);

figure;plot(x1,yr);

end

end