func [] = Muller(mat,n)  
coefficient = flipud(mat);  
root1 = input('Enter first root: ');  
root2 = input('Enter second root: ');  
root3 = input('Enter third root: ');  
maxRelError = input('Enter the maximum allowed relative approximate error(in %): ');  
convgCriteria = input('Enter convergence criterion for function value: ');  
maxIter = input('Enter the maximum number of iterations: ');  
error = 100;  
i = 0;  
while ((error >= maxRelError) && (i < maxIter))  
val1 = (polyval(coefficient, root3) - polyval(coefficient, root2))/(root3 - root2);  
val2 = (polyval(coefficient, root2) - polyval(coefficient, root1))/(root2 - root1);  
a = (val1 - val2)/(root3 - root1);  
b = val1 + a \* (root3 - root2);  
c = polyval(coefficient, root3);  
val3 = sqrt(b\*b - 4\*a\*c);  
ans1 = (-b + t3)/(2\*a);  
ans2 = (-b - t3)/(2\*a);  
if (abs(ans1) > abs(ans2))  
y = ans2;  
else  
y = ans1;  
end  
root1 = root2;  
root2 = root3;  
root3 = root3 + y;  
i = i + 1;  
err = abs((root3 - root2)/root3)\*100;  
e(i, 1) = error;  
if(abs(polyval(coefficient, root3)) < convgCriteria)  
break;  
end  
end  
if i >= maxIter  
disp('Maximum Iteration number attained.');  
elseif error <= maxRelError  
disp('Convergence for maximum relative approximate error reached.');  
else  
disp('Convergence criteria for function value reached.');  
end  
figure  
plot (1:i - 1, e(1:i - 1))  
grid on;  
title('MullerError vs Iteration')  
print -djpg muellerError.jpg

function y = f(x)  
y = mat(1,1);  
for i = 1:n  
y = y + mat(i+1,1)\*x.^i;  
end  
xn = linspace(root3 - 3, root3 + 3);  
yn = func(xn);  
figure  
plot(xn, yn);  
grid on;  
title('f(x) vs x')  
print -djpg muellerFunc.jpg  
end