Math 452, Numerical Methods:
Multi-step and Implicit Python Programs

```
from numpy import *
from matplotlib.pyplot import *
from matplotlib.patches import Patch

#f in the IVP y' = f(t,y), y(t0)=y0
def f(t,y):
    return (t-3.2)*y + 8*t*exp((t-3.2)**2/2)*cos(4*t**2)

def dfy(t,y):
    return t-3.2

#analytic solution to the IVP y' = f(t,y), y(t0)=y0
def sol(t,t0,y0):
    C = y0*exp(-(t0-3.2)**2/2)-sin(4*t0**2)
    return exp((t-3.2)**2/2)*(sin(4*t**2) + C)

#Backwards Euler (Implicit) Method
def beuler(t0,tn,n,y0):
    h = abs(tn-t0)/n
    t = linspace(0,tn,n+1)
    y = zeros(n+1)
    y[0] = y0
    for k in range(0,n):
        err = 1
        zold = y[k] + h*f(t[k],y[k]) #Use Forward Euler for initial guess
        I = 0
        #Use Newton's Method to solve implicit equation for y[k+1]
        while err > 10**(-10) and I < 5: #NM is limited to 5 iterations
            F  = y[k] + h*f(t[k+1],zold)-zold
            dF = h*dfy(t[k+1],zold)-1
            znew = zold - F/dF
            err = abs(znew-zold)
            zold = znew
            I +=1
        y[k+1] = znew
    return y
```

```
#Runge-Kutta "Classic" Order 4 method
def RK4(t0,tn,n,y0):
    h = abs(tn-t0)/n
    t = linspace(t0,tn,n+1)
    y = zeros(n+1)
    y[0] = y0
    for i in range(0,n):
        K1 = f(t[i],y[i])
        K2 = f(t[i]+h/2,y[i]+K1*h/2)
        K3 = f(t[i]+h/2,y[i]+K2*h/2)
        K4 = f(t[i]+h,y[i]+K3*h)
        y[i+1] = y[i] + h*(K1+2*K2+2*K3+K4)/6
    return y

#Adams-Bashforth 3 Step Method
def AdBash3(t0,tn,n,y0):
    h = abs(tn-t0)/n
    t = linspace(t0,tn,n+1)
    y = zeros(n+1)
    y[0:3] = RK4(t0,t0+2*h,2,y0)
    K1 = f(t[1],y[1])
    K2 = f(t[0],y[0])
    for i in range(2,n):
        K3 = K2
        K2 = K1
        K1 = f(t[i],y[i])
        y[i+1] = y[i] + h*(23*K1-16*K2+5*K3)/12
    return y

#Adams-Bashforth 3/Moulton 4 Step Predictor/Corrector
def PreCorr3(t0,tn,n,y0):
    h = abs(tn-t0)/n
    t = linspace(t0,tn,n+1)
    y = zeros(n+1)
    #Calculate initial steps with Runge-Kutta 4
    y[0:3] = RK4(t0,t0+2*h,2,y0)
    K1 = f(t[1],y[1])
    K2 = f(t[0],y[0])
    for i in range(2,n):
        K3 = K2
        K2 = K1
        K1 = f(t[i],y[i])
        #Adams-Bashforth Predictor
        y[i+1] = y[i] + h*(23*K1-16*K2+5*K3)/12
        K0 = f(t[i+1],y[i+1])
        #Adams-Moulton Corrector
        y[i+1] = y[i] + h*(9*K0+19*K1-5*K2+K3)/24
    return y
```

```
#Script to produce graphs
fg =1
n = 300
t0 = 0
tn = 6
y0 = .75
t = linspace(t0,tn,n+1)
ye = AdBash3(t0,tn,n,y0)
yb = beuler(t0,tn,n,y0)
ypc = PreCorr3(t0,tn,n,y0)
figure(fg)
plot(t,ye,'red',label='Adams-Bashforth 3')
plot(t,yb,'black',label='Backward Euler')
plot(t,yb,'cyan',label='Predictor/Corrector 3/4')
t = linspace(t0,tn,401)
ysol = sol(t,t0,y0)
plot(t,ysol,color ='green',label='Exact')
title('n = %d' % n)
axis([0,tn,-60,40])
legend(loc='lower left')
```