**ENGR 326**
**ODE Lab Assignment 2**

1. Determine if the following equations are exact. If they are exact, solve the equation.
   a. $(2xy+3)dx + (x^2-1)dy = 0$
   b. $(1/y)dx - (2y - x/y^2)dy = 0$
   c. $(\cos x \cos y + 2x)dx - (\sin x \sin y + 2y)dy = 0$
   d. $e^t(y-t)dt + (1+e^t)dy = 0$

2. One useful feature of Scilab is the built-in ODE solvers. Given a function that evaluates the derivatives (the differential equation) for specified values of the independent and dependent variables, a simple Scilab command will produce an output solution vector for a specified range of the independent variable values. The easiest form of the ODE solver command is

   `y=ode(y0,t0,t,f)`

   where

   `y` = vector or matrix of dependent variable values (the solution vector)
   `y0` = column vector of initial values for the dependent variable
   `t0` = initial value of the independent variable
   `t` = vector of independent variable values at which the dependent variable values are to be computed. One easy way to assign the values of this vector is to use the notation
      `t=[tstart:tincrement:tend]`
   where `tstart` is the first value of the independent variable at which a solution is to be computed, `tincrement` is the increment between successive values of the independent variable, and `tend` is the final value of the independent variable at which a solution is to be computed. For example,
      `t=[1:2:101]` would result in computing values at 1, 3, 5, 7, ... 99, 101.
   `f` = name of the function that evaluates the differential equation(s). For a relatively simple system of differential equations, an inline function can be used. An external function written in Fortran or C can also be used. The function must accept two arguments, `t` and `y`, where `t` is the scaler value of the independent variable, and `y` is a vector of the current estimate of the dependent variable values.

   If an inline function is to be used (as we will), it must be defined prior to using the `ode` command. A prototype function might look like the following

```
function dy = deriv(t,y)
  //function for 2 differential equations
  dy(1)=-t*y(1)/y(2)
  dy(2)=.3*t*y(2)
endfunction
```

The population dynamics of a predator and it's prey have been investigated by ecologists for centuries. One model developed early in the 20th century is Lotka-Volterra Two Species Model. Applied to a simple interaction between a fox community and a rabbit community yields the following system of differential equations

$$dR/dt = aR - bRF$$

$$dF/dt = ebRF - cF$$

where the parameters are defined by:

$a$ =natural growth rate of rabbits in the absence of predation (1/months)

$c$ =natural death rate of foxes in the absence of food (rabbits)(1/months)

$b$ =death rate per encounter of rabbits due to predation (1/fox-month)

$e$ =efficiency of turning predated rabbits into foxes (fox/rabbit)

solve this set of equations for 240 months using an initial population of 10,000 rabbits and 200 foxes. Assume that $a = 0.04/\text{month}$, $b = 0.0005/\text{fox-month}$, $c = 0.2/\text{month}$, and $e = 0.1$ fox/rabbit. You can plot the results using the Scilab plot2d command. A typical sequence of commands to plot both curves on the same axis might be as follows:

```
clf()       //Opens or clears the plot screen
plot2d(t,y(1,:),style=1)
plot2d(t,y(2,:),style=2)
xtitle(["Rabbit and Fox";"Interaction"],"Months","Population")
legends(["Rabbit","Fox"],style=[1,2])
```

(then click on the desired location for the legend in the graphics window)