FINAL: COMPUTER SCIENCE II

Date: **29th April 2016**
The Total points is **110** and the maximum you can score is **100** points.

(1) (5+5+8+7=25 points) Write down the output of the following commands in octave.
  (a) x=[1 2 3]; A=x'*x; y=A(2,:); disp(max(y));
  (b) A=[6 2 1;2 1 1; -1 0 1]; B=A(1:2,1:2); fprintf('Determinant of B is %d \n',det(B));
  (c) A=[6 2 1;2 1 1; -1 0 1]; b=[1 3 4]'; disp(inv(A)); disp(A\b);
  (d) x=[-pi:pi/4:pi]; plot(x,sin(x),"o-");

(2) (4+7+7+7=25 points) Describe what the following commands in octave do:
  (a) break
  (b) polyfit
  (c) A=[5 2; 2 1]; [l u p]=lu(A); disp(l); disp(u); disp(p);
  (d) ode45

(3) (10 points) The function $y = \frac{x}{c_1 x + c_2}$ can be transformed into a linear relationship $z = c_1 w + c_2$ with the change of variable $z = \frac{1}{y}$ and $w = \frac{1}{x}$. Write an "xlinxFit" function that calls linefit to fit data to $y = \frac{x}{c_1 x + c_2}$.

(4) (10 points) In the attached code for interpolation, identify the interpolation method and add comments in the code to explain the lines of the code which end with a % symbol. Also answer all the questions in mentioned there in the bold.

(5) (20 points) Let $\Phi_3(x) = \frac{1}{2}(5x^3 - 3x)$. Recall that $\Phi_3(x)$ is an orthogonal polynomial for the inner product $< f, g >= \int_{-1}^{1} f(t)g(t)dt$. Compute the weights for the Gaussian Quadrature method and use it to approximate $\int_0^2 e^{t^2} dt$.

(6) (20 points) Write down an octave function to find a solution to the differential equation

$$y' = e^{y+t} + sin(t), \; y(0) = 0$$

at t=2 using the stepsize h (which is a input variable for the function) following methods:
  (a) Midpoint method
  (b) Runge-Kutta method

```
function yhat = interpolate(x,y,xhat,fp1,fpn)
% Name the interpolation method
%
% Synopsis:   yhat = interpolate(x,y,xhat,fp1,fpn)
%
% Input:   x,y  = vectors of discrete x and y = f(x) values
%          xhat = (scalar or vector) x values where interpolant is evaluated
%          fp1  = slope at x(1), i,e., fp1 = f'(x(1))
%          fpn  = slope at x(n), i,e., fpn = f'(x(n));
%
% Output:  yhat = (vector or scalar) value(s) of the interpolant
%                 evaluated at xhat.  size(yhat) = size(xhat)

% --- Set up system of equations for b(i)
x = x(:);  y = y(:);  xhat = xhat(:);  %
n = length(x);
dx = diff(x);                  %
divdif = diff(y)./dx;          %

alpha = [0; dx(1:n-2); 0];                  %  sub diagonal
bbeta = [1; 2*(dx(1:n-2)+dx(2:n-1)); 1];    %  main diagonal
gamma = [0; dx(2:n-1); 0];                  %  super diagonal
A     = tridiags(n,bbeta,alpha,gamma);      %  Sparse, tridiagonal matrix
delta = [ fp1; ...
          3*(divdif(2:n-1).*dx(1:n-2) + divdif(1:n-2).*dx(2:n-1)); ...
          fpn ];

b = A\delta;                      %  Solve the system

% --- What does the following block do?
a = y(1:n-1);
c = (3*divdif - 2*b(1:n-1) - b(2:n))./dx;
d = (b(1:n-1) - 2*divdif + b(2:n))./dx.^2;
b(n) = [];                                  %


% --- Locate each xhat value in the x vector
i = zeros(size(xhat));  %  i is index into x such that x(i) <= xhat <= x(i+1)
for m=1:length(xhat)    %  For vector xhat: x( i(m) ) <= xhat(m) <= x( i(m)+1 )
  i(m) = binSearch(x,xhat(m));
end

% --- Nested, vectorized evaluation of the piecewise polynomials
xx = xhat - x(i);
yhat = a(i) + xx.*(b(i) + xx.*(c(i) + xx.*d(i)) );
```