```
function fe2dx_nd_fast_test ( )
%*****************************************************************************80
%
%% FE2DX_ND_FAST_TEST tests the FE2DX_ND_FAST code.
%
%  Discussion:
%
%    This function sets all parameter values and initial condition information
%    necessary to execute the "fast" version of the fe2dx_nd algorithm.
%
%  Licensing:
%
%    Copyright (C) 2014 Marcus R. Garvie.
%    See 'mycopyright.txt' for details.
%
%  Modified:
%
%    28 April 2014
%
%  Author:
%
%    Marcus R. Garvie.
%
%  Reference:
%
%    Marcus R Garvie, John Burkardt, Jeff Morgan,
%    Simple Finite Element Methods for Approximating Predator-Prey Dynamics
%    in Two Dimensions using MATLAB,
%    Submitted to Bulletin of Mathematical Biology, 2014.
%
  timestamp ( );
  fprintf ( 1, '\n' );
  fprintf ( 1, 'FE2DX_ND_FAST_TEST:\n' );
  fprintf ( 1, '  Test the FE2DX_ND_FAST function, which\n' );
  fprintf ( 1, '  applies Neumann and Dirichlet boundary conditions as it\n' );
  fprintf ( 1, '  approximates a solution to a predator-prey system.\n' );
%
%  Set the parameters.
%
  alpha = 0.4;
  beta = 2.0;
  gamma = 0.6;
  delta = 1.0;
%
%  Use T=150.0 for normal run.
%  Use T=0.50 for a "quick" run that might take 15 minutes of computing.
%
% T = 150.0;
  T = 0.50;
  delt = 1.0 / 384.0;
  t = tic;
  fe2dx_nd_fast ( alpha, beta, gamma, delta, T, delt, @u0f, @v0f, ...
    @g1uf, @g1vf, @g2uf, @g2vf );
  t = toc ( t );
  fprintf ( 1, '  Execution took %10.2g minutes \n', t / 60.0 );
```

```
%
%  Terminate.
%
  fprintf ( 1, '\n' );
  fprintf ( 1, 'FE2D_ND_FAST_TEST:\n' );
  fprintf ( 1, '  Normal end of execution.\n' );
  fprintf ( 1, '\n' );
  timestamp ( );
  return
end
function value = u0f ( x, y )
%*****************************************************************************80
%
%% U0F evaluates the initial condition for U.
%
%  Licensing:
%
%    Copyright (C) 2014 Marcus R. Garvie.
%    See 'mycopyright.txt' for details.
%
%  Modified:
%
%    26 April 2014
%
%  Author:
%
%    Marcus R. Garvie.
%
%  Parameters:
%
%    Input, real X, Y, a location in the region.
%
%    Output, real VALUE, the initial condition for U at (X,Y).
%
  value = 6.0 / 35.0 - 2.0E-07 * ( x - 0.1 * y - 225.0 ) * ( x - 0.1 * y - 675.0 );
  return
end
function value = v0f ( x, y )
%*****************************************************************************80
%
%% V0F evaluates the initial condition for V.
%
%  Licensing:
%
%    Copyright (C) 2014 Marcus R. Garvie.
%    See 'mycopyright.txt' for details.
%
%  Modified:
%
%    26 April 2014
%
%  Author:
%
%    Marcus R. Garvie.
%
%  Parameters:
%
```

```
%    Input, real X, Y, a location in the region.
%
%    Output, real VALUE, the initial condition for V at (X,Y).
%
  value = 116.0 / 245.0 - 3.0E-05 * ( x - 450.0 ) - 1.2E-04 * ( y - 150.0 );
  return
end
function value = g1uf ( x, y, t )
%*****************************************************************************80
%
%% G1UF evaluates the Dirichlet boundary condition for U.
%
%  Licensing:
%
%    Copyright (C) 2014 Marcus R. Garvie.
%    See 'mycopyright.txt' for details.
%
%  Modified:
%
%    28 April 2014
%
%  Author:
%
%    Marcus R. Garvie.
%
%  Parameters:
%
%    Input, real X, Y, a location on the boundary.
%
%    Input, real T, the time.
%
%    Output, real VALUE, the prescribed value for U at (X,Y,T).
%
  value = 0.0;
  return
end
function value = g1vf ( x, y, t )
%*****************************************************************************80
%
%% G1VF evaluates the Dirichlet boundary condition for V.
%
%  Licensing:
%
%    Copyright (C) 2014 Marcus R. Garvie.
%    See 'mycopyright.txt' for details.
%
%  Modified:
%
%    28 April 2014
%
%  Author:
%
%    Marcus R. Garvie.
%
%  Parameters:
%
%    Input, real X, Y, a location on the boundary.
```

```
170 %
171 %    Input, real T, the time.
172 %
173 %    Output, real VALUE, the prescribed value for V at (X,Y,T).
174 %
175   value = 0.0;
176   return
177 end
178 function value = g2uf ( x, y, t )
179 %*******************************************************************************80
180 %
181 %% G2UF evaluates the Neumann boundary condition for U.
182 %
183 %  Licensing:
184 %
185 %    Copyright (C) 2014 Marcus R. Garvie.
186 %    See 'mycopyright.txt' for details.
187 %
188 %  Modified:
189 %
190 %    28 April 2014
191 %
192 %  Author:
193 %
194 %    Marcus R. Garvie.
195 %
196 %  Parameters:
197 %
198 %    Input, real X, Y, a location on the boundary.
199 %
200 %    Input, real T, the time.
201 %
202 %    Output, real VALUE, the prescribed value for dU/dn at (X,Y,T).
203 %
204   value = 0.0;
205   return
206 end
207 function value = g2vf ( x, y, t )
208 %*******************************************************************************80
209 %
210 %% G2VF evaluates the Neumann boundary condition for V.
211 %
212 %  Licensing:
213 %
214 %    Copyright (C) 2014 Marcus R. Garvie.
215 %    See 'mycopyright.txt' for details.
216 %
217 %  Modified:
218 %
219 %    28 April 2014
220 %
221 %  Author:
222 %
223 %    Marcus R. Garvie.
224 %
225 %  Parameters:
226 %
```

```
227 %    Input, real X, Y, a location on the boundary.
228 %
229 %    Input, real T, the time.
230 %
231 %    Output, real VALUE, the prescribed value for dV/dn at (X,Y,T).
232 %
233   value = 0.0;
234   return
235 end
```

Published with MATLAB® R2013b