```
function fe2dx_nr_alt_fast_test ( )
%*****************************************************************************80
%
%% FE2DX_NR_ALT_FAST_TEST tests the FE2DX_NR_ALT_FAST code.
%
%  Discussion:
%
%    This function sets all parameter values and initial condition information
%    necessary to execute the "fast" version of the fe2dx_nr_alt 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_NR_ALT_FAST_TEST:\n' );
  fprintf ( 1, '  Test the FE2DX_NR_ALT_FAST function, which\n' );
  fprintf ( 1, '  applies Neumann and Robin 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;
  k1 = 0.01;
  k2 = 0.01;
  t = tic;
  fe2dx_nr_alt_fast ( alpha, beta, gamma, delta, T, delt, @u0f, @v0f, k1, ...
    k2, @g2uf, @g2vf );
```

```
  t = toc ( t );
  fprintf ( 1, '  Execution took %10.2g minutes \n', t / 60.0 );
%
%  Terminate.
%
  fprintf ( 1, '\n' );
  fprintf ( 1, 'FE2DX_NR_ALT_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.
%
```

```
113 %  Parameters:
114 %
115 %    Input, real X, Y, a location in the region.
116 %
117 %    Output, real VALUE, the initial condition for V at (X,Y).
118 %
119   value = 116.0 / 245.0 - 3.0E-05 * ( x - 450.0 ) - 1.2E-04 * ( y - 150.0 );
120   return
121 end
122 function value = g2uf ( x, y, t )
123 %*******************************************************************************80
124 %
125 %% G2UF evaluates the Neumann boundary condition for U.
126 %
127 %  Licensing:
128 %
129 %    Copyright (C) 2014 Marcus R. Garvie.
130 %    See 'mycopyright.txt' for details.
131 %
132 %  Modified:
133 %
134 %    28 April 2014
135 %
136 %  Author:
137 %
138 %    Marcus R. Garvie.
139 %
140 %  Parameters:
141 %
142 %    Input, real X, Y, a location on the boundary.
143 %
144 %    Input, real T, the time.
145 %
146 %    Output, real VALUE, the prescribed value for dU/dn at (X,Y,T).
147 %
148   value = 0.0;
149   return
150 end
151 function value = g2vf ( x, y, t )
152 %*******************************************************************************80
153 %
154 %% G2VF evaluates the Neumann boundary condition for V.
155 %
156 %  Licensing:
157 %
158 %    Copyright (C) 2014 Marcus R. Garvie.
159 %    See 'mycopyright.txt' for details.
160 %
161 %  Modified:
162 %
163 %    28 April 2014
164 %
165 %  Author:
166 %
167 %    Marcus R. Garvie.
168 %
169 %  Parameters:
```

```
%
%    Input, real X, Y, a location on the boundary.
%
%    Input, real T, the time.
%
%    Output, real VALUE, the prescribed value for dV/dn at (X,Y,T).
%
  value = 0.0;
  return
end
```

Published with MATLAB® R2013b