```
function [ret1,Y1,X2,Y2] = pdpts(n,xyrange)

%-------------------------------------------------------------------------------
% USAGE of "pdpts".
%
% Pad = pdpts(n)
% Pad = pdpts(n,xyrange)
% [X1,Y1,X2,Y2] = pdpts(n)
% [X1,Y1,X2,Y2] = pdpts(n,xyrange)
%
% Compute the (first family of) Padua points, either as a matrix Pad
% with their abscissas in the first column and their ordinates in the
% second, or as two subgrids X1,Y1 and X2,Y2, respectively.
%-------------------------------------------------------------------------------
% INPUT.
%
% n           : interpolation degree
% xyrange     : an optional vector [a,b,c,d] defining the rectangle
%               [a,b] x [c,d]. Otherwise, xyrange = [-1,1,-1,1]
%
% OUTPUT.
%
% Pad         : matrix of size ((n+1)*(n+2)/2) x 2 such
%               that (Pad(:,1),Pad(:,2)) defines the Padua points in the
%               rectangle [xyrange(1),xyrange(2)] x [xyrange(3),xyrange(4)].
% X1,Y1,X2,Y2 : the two subgrids X1,Y1 and X2,Y2 defining the Padua points
%-------------------------------------------------------------------------------

%-------------------------------------------------------------------------------
% FUNCTIONS CALLED BY THIS CODE:
% no external function is used.
%-------------------------------------------------------------------------------
% Authors:
%          Marco Caliari     <marco.caliari@univr.it>
%          Stefano De Marchi <demarchi@math.unipd.it>
%          Alvise Sommariva  <alvise@math.unipd.it>
%          Marco Vianello    <marcov@math.unipd.it>
%
% Date: November 2009.
%-------------------------------------------------------------------------------

if (nargin == 1)
% standard square [-1,1] x [-1,1]
% else rectangle [xyrange(1),xyrange(2)] x [xyrange(3),xyrange(4)]
  xyrange = [-1,1,-1,1];
end
zn = (xyrange(1)+xyrange(2)+(xyrange(2)-xyrange(1))*...
      cos(linspace(0,pi,n+1)))/2;
zn1 = (xyrange(3)+xyrange(4)+(xyrange(4)-xyrange(3))*...
       cos(linspace(0,pi,n+2)))/2;
if ((nargout == 0) || (nargout == 1))
% points as a single matrix
  [Pad1,Pad2] = meshgrid(zn,zn1);
  findM = [2:2:(n+1)*(n+2)]';
  if (mod(n,2) == 0)
    add = repmat([0,1],(n+2)/2,n/2);
    add = [add,zeros((n+2)/2,1)];
    findM = findM-add(:);
  end
%  [M1,M2] = meshgrid([0:n],[0:n+1]);
%  findM = find(mod(M1+M2,2));
  Pad = [Pad1(findM),Pad2(findM)];
  ret1 = Pad;
else
% points as two (mesh)grids
  En = zn(1:2:n+1);
  On = zn(2:2:n+1);
  En1 = zn1(1:2:n+2);
  On1 = zn1(2:2:n+2);
  [X1,Y1] = meshgrid(En,On1);
  [X2,Y2] = meshgrid(On,En1);
  ret1 = X1;
end

%--------------------------------------------------------------------------------
% OCTAVE TESTS.
%--------------------------------------------------------------------------------
% Octave testing: type
%
% test pdpts
%
% at the Octave prompt
%
%!test
%! disp('Degree 0')
%! Pad = pdpts(0);
%! expected = [-1,-1];
%! assert(Pad,expected,10*eps);
%!test
%! disp('Degree 1')
%! Pad = pdpts(1);
%! expected = [cos([0;1;1]*pi),cos([1;0;2]*pi/2)];
%! assert(Pad,expected,10*eps);
%!test
%! disp('Degree 2 with xyrange')
%! Pad = pdpts(2,[0,1,0,2]);
%! expected = [(cos([0;0;1;1;2;2]*pi/2)+1)/2,cos([1;3;0;2;1;3]*pi/3)+1];
%! assert(Pad,expected,10*eps);
```