```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% lagrange_interp.m
%-------------------------------------------------------------------------
% C Rocheleau, Colorado State University
% 9/23/23
%-------------------------------------------------------------------------
% This function performs Lagrange's method to create an interpolating
% polynomial from a list of given points and evaluates at points X
%-------------------------------------------------------------------------
% INPUTS
%   x_given: A vector of X positions of known points to use to find the
%       interpolating polynomial
%   y_given: A vector of Y positions of known points to use to find the
%       interpolating polynomial
%   X: Points at which to evaluate the interpolating polynomial
%-------------------------------------------------------------------------
% OUTPUTS
%   Y: Output of interpolating polynomial at given points
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Y = lagrange_interp(x_given, y_given, X)

Y = zeros(size(X));

% Iterate across the Y balues of our given points
for iY = 1:length(y_given)
    % Reset our numerators and denominators
    tempnum = 1;
    tempden = 1;
    % Construct our numerator and denominators by iterating across measx
    for iX = 1:length(x_given)
        if iX ~= iY     % Don't use the point we are at to avoid 0/0
            tempnum = tempnum.*(X - x_given(iX));
            tempden = tempden.*(x_given(iY) - x_given(iX));
        end
    end
    % Add our newest term onto our outputs
    Y = Y + y_given(iY).*tempnum./tempden;
end
```

```
Not enough input arguments.
Error in lagrange_interp (line 23)
Y = zeros(size(X));
```

Published with MATLAB® R2022b