Straight Line Function

```
% this function plots points between a starting point and an ending point
% the starting point is defined by x1, y1
% the ending point is defined by x2, y2
% 1/ptsPerMeter (points per meter) is the increment used to plot the
% points along the x-axis (Case I) or y-axis (Case II) - see graphic
function xy = StraightLine(x1, y1, x2, y2, ptsPerMeter)
  x = [0 0 0];                                    % default values for the vector x
  y = [0 0 0];                                    % default values for the vector y

  if x1 < 0 || y1 < 0 || x2 < 0 || y2 < 0         % starting and ending points must be in Quardrant I
    disp("""StraightLine"" function arguments ""x1, y1, x2, y2"" must be >= 0")

    if x1 < 0                       % if x1 < 0
      fprintf('x1 = %d\n', x1);     % display "X1 = (value of x1)"
    end

    if y1 < 0                       % if y1 < 0
      fprintf('y1 = %d\n', y1);     % display "y1 = (value of y1)"
    end

    if x2 < 0                       % if x2 < 0
      fprintf('x2 = %d\n', x2);     % display "X2 = (value of x2)"
    end

    if y2 < 0                       % if y2 < 0
      fprintf('y2 = %d\n', y2);     % display "y2 = (value of y2)"
    end

  elseif ptsPerMeter <= 0                         % points per meter must be greater thatn zero
    disp("""StraightLine"" function argument ""ptsPerMeter"" must be > 0")
    fprintf('ptsPerMeter = %d\n', ptsPerMeter);     % display "ptsPerMeter = (value of ptsPerMeter)"
  elseif (x1 == x2) && (y1 == y2)                 % starting and ending points can not be the same point
    disp("""StraightLine"" function arguments ""x1 = x2 and y1 = y2""")
    disp("Starting point and ending point can not be the same point")
    fprintf('x1 = x2 = %d\n', x1);                % display "x1 = x2 = (value of x1)"
    fprintf('y1 = y2 = %d\n', y1);                % display "y1 = y2 = (value of y1)"
  else
    increment = 1/ptsPerMeter;                    % calculate the increment used to plot the line
    slope = (y2 - y1) / (x2 - x1);                % calculate the slope of the line
```

Straight Line Function

```
% Case I - for lines with a slop less than or equal to 1 and greater than or equal to -1
% we will measure increments along the x-axis
if abs(slope) <= 1                                % if the slope qualify as a Case I slope
  if (increment > abs(x2 - x1))                   % check the length of the increment
    increment = abs(x2 - x1);                     % it can not be greater than the
    disp("ptsPerMeter is to small")               % the horizontal distance between the
    disp("therefore increment was to large")      % starting point and the ending point
    disp("increent was reduced to abs(x2-x1)")    % if it is, set the increment equal to
  end                                             % that horizontal distance

  if x2 < x1                                      % if we are drawing the line from right to left
    increment = -increment;                       % the increment muat have a negative value
  end

  % re-develop the x vector
  % we want the last value of the x vector to be equal to x2
  % initially the last value of the x vector may or may not be equal to x2
  % adding x2 to the x vector will assure ourselves that the last value is x2
  % if the last value is already x2, adding one more x2 to the
  % x vector won't hurt anything

  x = x1:increment:x2;
  x = [x, x2];
  y = y1 + ((x - x1) * slope);      % for each point on the x axis, calculate a point on the y axis
else
  % Case II - for lines with a slop greater than 1 or less than -1
  % we will measure increments along the y-axis

  if (increment > abs(y2 - y1))                     % check the length of the increment
    increment = abs(y2 - y1);                       % it can not be greater than the
    disp("ptsPerMeter is to small")                 % the vertical distance between the
    disp("therefore increment was to large")        % starting point and the ending point
    disp("increent was reduced to abs(y2-y1)")      % if it is, set the increment equal to
  end                                               % that vertical distance

  if y2 < y1                                        % if we are drawing the line from top to bottom
    increment = -increment;                         % the increment muat have a negative value
  end
```

Straight Line Function

```
      % re-develop the y vector
      % we want the last value of the y vector to be equal to y2
      % initially the last value of the y vector may or may not be equal to y2
      % adding y2 to the y vector will assure ourselves that the last value is y2
      % if the last value is already y2, adding one more y2 to the
      % y vector won't hurt anything

      y = y1:increment:y2;
      y = [y, y2];
      x = x1 + ((y - y1) * (x2 - x1) / (y2 - y1));      % for each point on the y axis, calculate a point on the x
axis
    end
  end

  xy = [x', -y'];                                       % develop an xy matrix based on the x and y vectors
end
```