```
function minima = quad_3pt_refine(x0, step_size)

[x(1), x(3)] = bounding_phase_algo(x0, step_size);
x(2) = (x(1) + x(3))/2;
% Defining the control points

x = [x(1), x(2), x(3)]
% storing the values in a matrix form

f = [objF(x(1)), objF(x(2)), objF(x(3))]
% f matrix

Ex = 0.0001;
% Ef = 0.001;
% Defining the convergence criteria

disp('Start of iterations');

while true

    [fmin, i] = min(f);
    xmin = x(i);
    % Finding fmin and corresponding xmin

    a0 = f(1)
    a1 = (f(2)-f(1))/(x(2)-x(1))
    a2 = (((f(3)-f(1))/(x(3)-x(1))) - a1)/(x(3)-x(2))
    % coefficient values

    x_t = (a2*(x(1)+x(2)) - a1)/(2*a2)
    f_t = objF(x_t)
    % x~ and f~ values calculations

    if (abs((x_t - xmin)/xmin) <= Ex) % && ((f_t - fmin)/fmin <= Ef)
        minima = x_t;
        break
        % Checking the convergence criteria
    else
        x = [x, x_t];
        x = sort(x)
        % Adding x~ to the x matrix

        f = [f, f_t];
        for j = 1:4
            f(j) = objF(x(j));
        end
        % f matrix
        f
```

```
        [~, j] = min(f);
        x2 = x(j);
        x1 = x(j-1);
        x3 = x(j+1);
        x(4) = [];
        x = [x1, x2, x3]
        f = [objF(x1), objF(x2), objF(x3)]
        % Updated x and f matrix
        continue
    end

end

end
```