```
function [l2fehler,h1fehler] =
fe_fehler(vertices,triangles,funchandle,u)

[Qp,Qw] = fe_quadrature(3);    % Referenz-Quadratur
Qsize = size(Qw,2);
[Sf, SDf] = fe_shapes(Qp);     % Referenz-Formfunktionen an den
                               % Quadraturpunkten

l2fehler=0;
h1semi=0;

for m=1:M                      % Schleife ueber Dreiecke
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Hier: Bestimme Element-Transformation, d.h. Fm,Jm,dm,Fm^{-T}
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for q=1:Qsize              % Schleife ueber Quadraturpunkte

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % Loese lineare Interpolationsaufgabe um Uh, DUh zu bestimmen
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        Uh = 0;                % Fuer Auswertung der Finite-Element
        DUh = 0;               % Funktion an den Quadraturpunkten
        for i=1:3

            Uh = Uh + ...

            DUh = DUh + ...
        end
     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     % Exakter Funktionswert bzw. Ableitung, sowie Fehler berechnen
     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        [U_ex,DU_ex] = ....    % Berechnung ueber funchandle
        error_u = ...
        error_Du = ...

        l2fehler = l2fehler + ...

        h1semi = h1semi + ...
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Hier: L2fehler bzw. H1fehler berechnen
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function [A,b] = fe_assemble(vertices,edges,triangles,f_hdl,g_hdl)

[Qp,Qw] = fe_quadrature(1);    % Referenz-Quadratur
Qsize = size(Qw,2);
[Sf, SDf] = fe_shapes(Qp);     % Referenz-Formfunktionen an den
                               % Quadraturpunkten

%% Speicher und Zaehler fuer Sparse-Matrix, Loesung und rechte Seite
cnt=0;
dummysize=100;
row = zeros(1,dummysize*N);
column = zeros(1,dummysize*N);
data= zeros(1,dummysize*N);
ell=zeros(N,1);

for m=1:M                      % Schleife ueber Dreiecke
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Hier: Bestimme Element-Transformation, d.h. Fm,Jm,dm,Fm^{-T}
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for q=1:Qsize              % Schleife ueber Quadraturpunkte
        % aktueller Quadraturpunkt
        x_q = ...

        for i=1:3              % Schleife fuer \phi_z
            Index_z = triangles(m,i);

            ell(Index_z) = ell(Index_z) + ...

            for k=1:3          % Schleife fuer \phi_y
                Index_y = triangles(m,k);
                cnt=cnt+1;
                row(cnt) = Index_z;
                column(cnt) = Index_y;

                data(cnt) = ...
            end
        end
    end
end
%% Konstruktion der Matrix und innere Punkte
K = sparse(row(1:cnt),column(1:cnt),data(1:cnt),N,N);
RandV = unique(edges);
InnereV = setdiff(1:N,RandV);
v = zeros(N,1);

% Randpunkte
v(RandV) = ....

b = ell(InnereV) - K(InnereV,RandV)*v(RandV);
A = K(InnereV,InnereV);
```