```
% ------------------ Function my_rk2implicit.m  ----------------
% This function implements second order semi-implicit
%  Runge-Kutta algorithm
% fun_name: Name of function that returns derivatives
% vector F(x,t) given x and t
% xn : initial value of the state vector
% h : integration step size
% eps:  tolerence
% maxiter: maximum number of iterations
% ------------------------------------------------------------------

function [xnplus1, converged ] = my_rk2implicit( fun_name, xn,
tn, h, eps, maxitr )

% Heun's modified rule is used for initialization of iterations

K1 = feval( fun_name, tn, xn ) ;
xn1 = xn + h * K1  ;
K2 = feval( fun_name, tn+h, xn1 ) ;
xinit = xn + 0.5*h*( K1 + K2 ) ;
xt = xinit ;

% Iterations for semi-implicit RK method

delta = 100 ; itr = 0 ;

while ( (delta > eps) && (itr < maxitr ))
   xn1 = 0.5*(xinit+xn) ;
   K2 = feval( fun_name, tn+(h/2),  ) ;
   xnew = xn + h * K2 ;
```

```
    xinit = xnew ;      % set last x as new iteration guess
   % Convergence criterion
    delta = norm( xnew - xinit ) / norm(xnew) ;
    itr = itr+1 ;
end

if (itr < maxitr )
    converged = 1 ;
else
    converged = 0 ;
end

xnplus1 = xnew
```