```
%-------------------------------------------------------------
% Tony Hyun Kim
% CS 246, PS #3, Problem 3(g)
%-------------------------------------------------------------
clear all;

% Load the graph from the text file
source = 'graph.txt';
edges = load(source);
m = length(edges(:,1));

n = 100; % Number of nodes
global P;
P = zeros(n,n);

% Fill in the edges
for i = 1:m
    edge = edges(i,:);
    P(edge(1),edge(2)) = 1;
end

% Now renormalize the rows of P by degree
for i = 1:n
    p = P(i,:);
    P(i,:) = p/sum(p);
end

% Run Power Iteration
%-------------------------------------------------------------
global eps;
eps = 0.2;
Niter = 40;
pis = zeros(1+Niter,n);
pi0 = 1/n*ones(1,n);
pis(1,:) = pi0;

ts = zeros(1,Niter);
for j = 1:Niter
    tic;
    pis(1+j,:) = eps*pi0+(1-eps)*pis(j,:)*P;
    ts(j) = toc;
end
fprintf('Power iteration: %d iterations, %.3f ms per iter, %.3f ms total\n',...
    Niter,mean(ts)*1e3,sum(ts)*1e3);

pi = pis(end,:); % The (approximate) PageRank distribution

% Sort the nodes
[~, rank] = sort(pi,'descend');

% MC3 algorithm
%-------------------------------------------------------------
R = 5;

pi_mc3 = zeros(1,n);

count = 0;
time = 0;
for i = 1:n % Fix initial node
    for j = 1:R % Iterations
        tic;

        % Random surfer (MC3)
        xi = i;
        pi_mc3(xi) = pi_mc3(xi)+1;
        while(rand>eps) % Keep on surfing
           p = cumsum(P(xi,:));
           xi = find(rand<p,1);
```

```
            pi_mc3(xi) = pi_mc3(xi)+1;
        end

        time  = time + toc;
        count = count + 1;
    end
end
fprintf('MC3 with R=%d: %.3f ms per walk, %.3f ms total\n',...
    R,time/count*1e3,time*1e3);

% Normalize result
pi_mc3 = eps/(n*R)*pi_mc3;

% Compute errors
for c = [10 30 50 100]
    e = abs(pi(rank(1:c))-pi_mc3(rank(1:c)));
    fprintf('Top-%d error: %.3f\n',c,mean(e));
end

% MC2 algorithm
%-----------------------------------------------------------
R = 5;

pi_mc2 = zeros(1,n);

count = 0;
time = 0;
for i = 1:n % Fix initial node
    for j = 1:R % Iterations
        tic;

        % Random surfer (MC2)
        xi = i;
        while(rand>eps) % Keep on surfing
            p = cumsum(P(xi,:));
            xi = find(rand<p,1);
        end
        pi_mc2(xi) = pi_mc2(xi) + 1;

        time  = time + toc;
        count = count + 1;
    end
end
fprintf('MC2 with R=%d: %.3f ms per walk, %.3f ms total\n',...
    R,time/count*1e3,time*1e3);

% Normalize result
pi_mc2 = 1/(n*R)*pi_mc2;

% Compute errors
for c = [10 30 50 100]
    e = abs(pi(rank(1:c))-pi_mc2(rank(1:c)));
    fprintf('Top-%d error: %.3f\n',c,mean(e));
end

% MC1 algorithm
%-----------------------------------------------------------
R = 5;
N = n*R;

pi_mc1 = zeros(1,n);

count = 0;
time = 0;
for i = 1:N
    tic;

    % Random surfer (MC1)
```

```
    xi = randi(n);
    while(rand>eps) % Keep on surfing
       p = cumsum(P(xi,:));
       xi = find(rand<p,1);
    end
    pi_mc1(xi) = pi_mc1(xi) + 1;

    time  = time + toc;
    count = count + 1;
end
fprintf('MC1 with R=%d: %.3f ms per walk, %.3f ms total\n',...
    R,time/count*1e3,time*1e3);

% Normalize result
pi_mc1 = 1/(n*R)*pi_mc1;

% Compute errors
for c = [10 30 50 100]
    e = abs(pi(rank(1:c))-pi_mc1(rank(1:c)));
    fprintf('Top-%d error: %.3f\n',c,mean(e));
end
```