```
function IT=JIt(A,b,tol);
%performs Jacobian iteration on linear system;
%Inputs:
%  A:   n-by-n matrix
%  b:   1-by-n RHS vector
%  tol: stop when ||x_n+1 - n_x|| <tol (use infinite norm)
%  Output:
%  IT is the number of iterations for convergence;
n=size(A); n=n(1); %determine dimension of A
x0(1:n,1)=0; %Set initial guess to 0 vector;
D=zeros(n); %Build the Diagonal Matrix;
for i=1:n;
    D(i,i)=A(i,i);
end
L=zeros(n); %Build the L matrix
for i=1:n;
    for j=1:n;
        if i>j;
            L(i,j)=-A(i,j);
        end
    end
end
U=-(A-D+L); %Build the U matrix
T=inv(D)*(L+U); %Create the T matrix
[v,d]=eig(T); %find the spectral radius, determine if Jacobe system can converge;
d=(d*conj(d))^.5;
if max(max(d))>=1;
    disp(['r(T)=',num2str(max(max(d))),'>1, Can not set up iterative system that will converge to a solution']);
    return;
end
c=inv(D)*b; %modify right hand side vector;
data=([0,x0',0]); %build a data matrix;
for i=1:100; %start iterations
    x1=T*x0+c;
    data=[data;[i,x1',norm((x1-x0),inf)]];
    if norm((x1-x0),inf)<tol, %quit if below tolerance;
        IT=i;
        break
    end
    x0=x1;
end
%Output
disp(' ');
disp('Iteration # // x1 through xn // ||x_(k+1)-x_(k)||');
disp(num2str(data)); %dump data table
space(1:n,1:n)=' ';%blank space for display of L, D, U
disp(' ');
disp('A/L/D/U');
disp([num2str(A),space,space,num2str(L),space,space,num2str(D),space,space,num2str(U)]);
disp(' ');
disp('T');
disp(num2str(T));
```