```
%SOLVER----Risolve il problema memorizzando le variabili di interesse
%----------non produce analisi grafiche


%    Risoluzione del flusso in cavita' quadrata nelle variabili psi-vor

%   Discretizzazione del dominio precedentemente effettuata attraverso
%                       il programma "meshatura"

%               Elaborato sviluppato da ZONTA FRANCESCO

%-------------------------------------------------------------------------

%%%%%%%%%  Griglia di clcolo
L=1;
H=1;

nl=100;
nh=100;


dx=L/nl;
dy=H/nh;


%tabella x
x=zeros((nl+1)*(nh+1), 4);
yc=0;
nodo=0;
for j=1:nh+1
    xc=0;
    for i=1:nl+1
        nodo=nodo+1;
        x(nodo,:)=[i j xc yc];
        xc=xc+dx;
    end
    yc=yc+dy;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%Parametri e variabili fondamentali

Re=100;                 %Numero di Reynolds
dt=10^-3;               %intervallo di tempo



u=zeros(nl+1,nh+1);      %componente u della velocit@
v=zeros(nl+1,nh+1);      %componente v della velocit@

psi=zeros(nl+1,nh+1);    %funzione di corrente

vor=zeros(nl+1,nh+1);    %vorticit@
vor2=zeros(nl+1,nh+1);   %vorticit@ all'istante successivo

Nconv=0;                 %percentuale dei nodi che raggiungono la stazionariet@(convergenza)

%----------------------------

%condizioni al contorno

for i=1:nl+1
     u(i,nh+1)=1;
end


%-------------------------
eps=10^-6;               %errore massimo tollerabile
time=0;                  %contatore temporale
timetot=50;              %tempo massimo di simulazione
```



```
NODI=(nl+1)*(nh+1);      %numero totale di nodi
numero=0;                %numero di nodi convergenti
itertot=0;               %numero di iterazioni totali

%-------------------------------------------------------------------------

while numero<NODI & time<timetot

    time=time+dt
    itertot=itertot+1;

 %Calcolo della funzione di corrente (noto il campo di vorticit@)
 %             attraverso l'equazione di Poisson
 %          per una meshatura equispaziata su x e y


w=1.9;                        %peso dell'algoritmo SOR
itermax=1000;
iter=0;
num=0;
numeq=(nl-1)*(nh-1);           %numero di equazioni considerate
while num<numeq & iter<itermax
    num=0;

    for i=2:nl
      for j=2:nh
       psi2=(psi(i+1,j)+psi(i-1,j)+psi(i,j+1)+psi(i,j-1)-vor2(i,j)*dx^2)/4;
       psi2=(1-w)*psi(i,j)+w*psi2;
       error1=abs(psi2-psi(i,j));
       psi(i,j)=psi2;

       if error1<10^-6
           num=num+1;
       end
      end
    end
    iter=iter+1;
end



%Calcolo della vorticit@ al bordo espandendo in serie di Taylor la psi al
%2@ ordine


    %bordi verticali
    for j=1:nh+1
        vor2(1,j)=(2/dx^2)*psi(2,j);                    %bordo  sinistro
        vor2(nl+1,j)=(2/dx^2)*psi(nl,j);                %bordo destro
    end

    %bordi orizzontali
    for i=1:nl+1
        vor2(i,1)=(2/dy^2)*psi(i,2);                    %bordo basso
        vor2(i,nh+1)=(2/dy^2)*(psi(i,nh)+dy*u(2,nh+1)); %bordo alto
    end


                                %La matrice della vorticit@ (vor) al passo
                                %n viene sostituita da quella all'istante
for i=1:nl+1                    %finale n+1,detta vor2, per affinarne
    for j=1:nh+1                %la precisione
        vor(i,j)=vor2(i,j);
    end
end


  %Calcolo del campo di moto (attraverso la nuova psi ricavata da Poisson)
  for i=2:nl
      for j=2:nh
          u(i,j)=(psi(i,j+1)-psi(i,j-1))/(2*dy);
          v(i,j)=-(psi(i+1,j)-psi(i-1,j))/(2*dx);
```

```
      end
  end


%        Calcolo della vorticit? nel nucleo del dominio
%  all'istante successivo attraverso l'equazione del trasporto

     for i=2:nl
         for j=2:nh

a(i,j)=(0.5/dx)*(u(i+1,j)*vor(i+1,j)-u(i-1,j)*vor(i-1,j))+(0.5/dy)*(v(i,j+1)*vor(i,j+1)-v(i,j-1)*vor
(i,j-1));
b(i,j)=(vor(i+1,j)-2*vor(i,j)+vor(i-1,j))/(dx^2)+(vor(i,j+1)-2*vor(i,j)+vor(i,j-1))/(dy^2);

          vor2(i,j)=vor(i,j)+ dt*(-a(i,j)+(1/Re)*b(i,j));
         end
     end



% Calcolo dell'errore sulla vorticit?

numero=0;
for i=1:nl+1
    for j=1:nh+1
        error=abs(vor2(i,j)-vor(i,j));

                                    %  Condizione per incrementare
                                    % il numero di nodi che raggiungono
                                    % la stabilit?(nel ciclo while)

         if error<eps
            numero=numero+1;
         end
                                     %salva il valore degli errori dei punti
         if i==j                     %sulla diagonale del quadrato
            f_err(itertot,i)=error;  %in un vettore per tracciarne l'andamento

         end

end

end
N=numero;

end


%---------------------------------

for i=1:nl+1                         %calcolo del modulo della velocit?
    for j=1:nh+1
        vel(i,j)=(u(i,j)^2+v(i,j)^2)^0.5;
    end
end

%--------------------------------------------------------------------------

%        Visualizzazione dei risultati numerici


fprintf('\n\n tempo di simulazione \n\n');
time

fprintf('\n\n valori della vorticit? \n\n');
vor;

fprintf('\n\n velocit? lungo x \n\n');
u;

fprintf('\n\n velocit? lungo y \n\n');
```



```
v;

fprintf('\n\n numero di nodi che hanno raggiunto stazionariet? \n\n')
N                    %numero di nodi che soddisfano la convergenza

fprintf('\n\n percentuale dei nodi in stazionariet? \n\n')
Nconv=(N/NODI)*100    %percentuale di nodi convergenti

fprintf('\n\n modulo della velocit? \n\n');
vel;

Reynolds=Re

%--------------------------------------------------------------------------
```