1. Blasius similarity equation is written as

$$\frac{d^3 f}{d\eta^3} + \frac{f}{2}\,\frac{df^2}{d\eta^2} = 0 \text{ , with f (0) = 0, f'(0) = 0, f'(inf) = 1}$$

Transformation into first order ODEs

$$\frac{df}{d\eta} = g$$
$$\frac{dg}{d\eta} = h$$
$$\frac{dh}{d\eta} + \frac{1}{2} f.h = 0$$

After, discretizing on forward difference method,

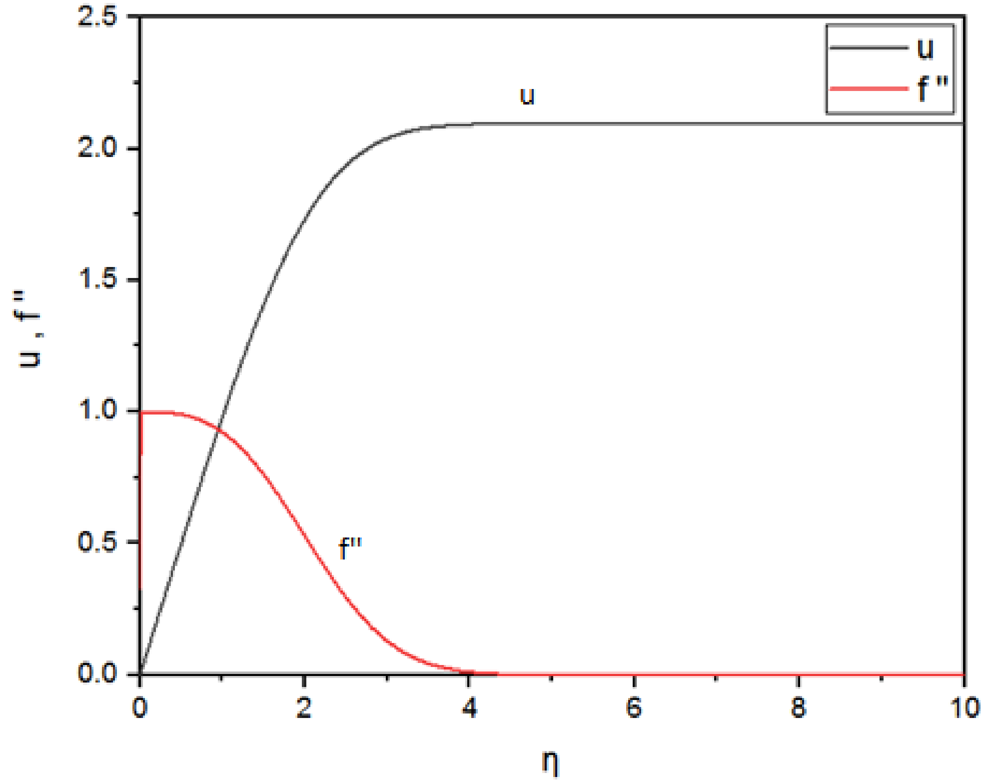
f (i+1) = f (i) + $\Delta\eta\ g(i)$
g(i+1) = f (i) + $\Delta\eta\ h(i)$
h(i+1) = h (i) - $\frac{1}{2}\Delta\eta\ f(i).h(i)$

```
% Matlab coding for Blasius similarity equation and similarity energy equation

clc;
clear all;

% Blasius similarity equation: f''' + (1/2)*f*f''=0 with f(0) = 0, f'(0) = 0, f'(inf) = 1
% Transforming Blasius Equation into a system of first-order ODEs:
%       df/d(eta)= g
%       dg/d(eta)= h
%       dh/d(eta)+(1/2)*f*h = 0

% f=y1
% g=y2
% h=y3
% d(eta)=dn

%% Parameters of Blasius Equation
    U_inf = 1;                 %free stream velocity
    L = 10;                    %length of flat plate
    mu = 1.789E-5;             %viscosity of air
    rho = 1.225;               %density of air
    nu = mu/rho;               %kinematic viscosity
    A = sqrt(nu/U_inf);
% total number of nodes
    N=1000;
% initial conditions
    dn = 0.01;
    y1(1) = 0;
    y2(1) = 0;
    y3(1) = 1;                   % 1st Guess

% discretizing on forward difference method
for i = 1:N
        y3(i+1) = y3(i) - (1/2)*y1(i)*y3(i)*dn;
        y2(i+1) = y3(i)*dn + y2(i);
        y1(i+1) = y2(i)*dn + y1(i);
```

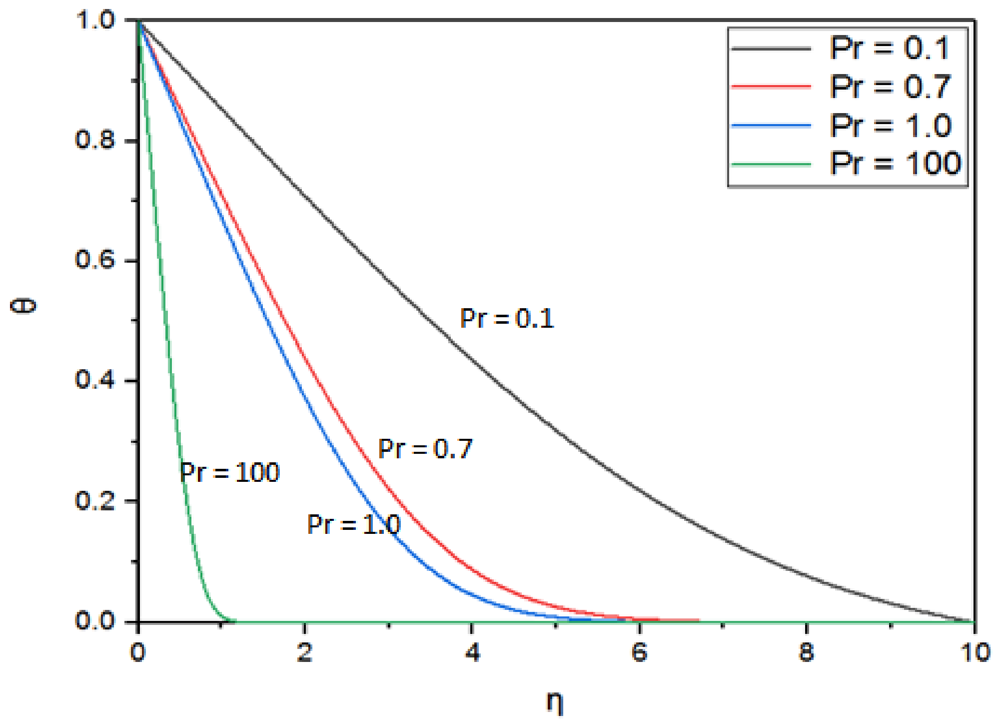
```
end
        y2_end=y2(N);

y2_new(1) = 0;
y3_new(1) = 1.1;                % 2nd Guess

for i = 1:N
        y3_new(i+1) = y3_new(i) - (1/2)*y1(i)*y3_new(i)*dn ;
        y2_new(i+1) = y2_new(i) + y3_new(i)*dn;
        y1(i+1) =  y1(i) + y2_new(i)*dn ;
end
        y2_newend=y2_new(N);

while A > 1e-9                  %error
      % Using 'Newton Raphson' Method
          y = y3_new(1)- (y2_newend-1)*(y3_new(1)-y3(1))/(y2_newend-y2_end);
          y3(1) = y3_new(1);
          y3_new(1) = y;
          y2_end=y2_newend;

            for i = 1:(N)
                y3_new(i+1) = y3_new(i) - ((1)/2)*y1(i)*y3_new(i)*dn;
                y2_new(i+1) = y3_new(i)*dn + y2_new(i);
                y1(i+1) = y2_new(i)*dn + y1(i);
            end

    y2_newend=y2_new(N);
    A = abs(y2_newend-1);        %error estimation
end
eta = 0:dn:N*dn;

% Result
    eta=[eta]';
    f=[y1]';
    g=[y2]';
    h=[y3]';
finalsolution = table(eta, f, g, h)

% Energy equation: theta'' + (f/2)*pr*theta'=0 with theta(0)=1, theta(inf)=0
% Transforming into a system of first-order ODEs:
%          d(theta)/d(eta)= x
%          dx/d(eta)= -(1/2)*pr*f*x

pr = 100;                       %Prandtl number

% Initial condition
th1(1)=1;

x1(1)=1;                        %1st guess for energy equation
r= 0.1;
% discretizing on forward difference method
for i=1:N
        x1(i+1) = x1(i)-(1/2)*pr*y1(i)*x1(i)*dn;
        th1(i+1) = th1(i)+x1(i)*dn;
end
```

```
    th1_end = th1(N);

 th2(1)=1;
 x1_new(1)=1.1;                 %2nd guess for energy equation

for i=1:N
        x1_new(i+1) = x1_new(i)-(1/2)*pr*y1(i)*x1_new(i)*dn;
        th2(i+1) = th2(i)+x1_new(i)*dn;
end
    th2_end = th2(N);

while r>1e-9                         %error
    % using 'Newton-Raphson' Method
        x = x1_new(1)-(th2_end*(x1_new(1)-x1(1)))/(th2_end-th1_end);
        x1(1)=x1_new(1);
        x1_new(1)=x;th1_end = th2_end;

            for i=1:N
                x1_new(i+1) = x1_new(i)-(1/2)*pr*y1(i)*x1_new(i)*dn;
                th2(i+1) = th2(i)+x1_new(i)*dn;
            end

    th2_end = th2(N);
    r = abs(th2_end);               % error estimation
end

% output

figure(1);
    plot (eta,y2_new,'r','LineWidth',2)
    hold on
    plot (eta,y3_new,'k','LineWidth',2)
    legend('f^{(1)}','f^{(2)}')
    legend boxoff;
    xlabel('\eta','FontSize',15);
    ylabel('\it{ f^{(1)}, f^{(2)}}','FontSize',15);

figure(2);
    eta = 0:dn:N*dn;
    plot (eta,th2,'b','LineWidth',2);
    legend('Pr=100')
    legend boxoff;
    xlabel('\eta','FontSize',15);
    ylabel('\theta','FontSize',15);

%end
```

Plot: u vs η and f'' vs η



Plot: θ vs η



Result:

S. No	Pr	$\theta'(0)$
1	0.1	-0.146
2	0.7	-0.292
3	1	-0.330
4	100	-1.55

2.
```
%Matlab coding for Nusselt Number

clc;
clear all;

N= 2000;                %total nodes
Nu = 0.5;               %initial guess for Nu
dr = 0.001;

%Initial condition
th(1) = 0;
r(1) = 0;

err= 0.1;

while err>1e-9          %error

    Nu = Nu + 0.01;
    f(1) = Nu/2;

%discretization on backward difference method

    for i = 2:N
        r(i) = r(i-1) + dr;
        f(i) = f(i-1) + dr*( f(i-1)/(1-r(i-1)) -(2*Nu*(1-(1-r(i-1))^2)*th(i-1)));
        th(i) = f(i-1)*dr + th(i-1);
    end

    err = f(N);         % error calculation

end

display(Nu)

% end
```

Result:

Nu = 3.6700