```
% Bisection method %
%function definition%
    f = @(x)x^4-2*x^3-4*x^2+4*x+4;
%initial conditions%
    a=-2;
    b=-1;
    c = (a+b)/2;
    eps = 10^(-2);
%root approximation%
    while abs(b-c)>=eps
        if sign(f(b))*sign(f(c))<=0
            a = c;
        else
            b = c;
        end
        c=(a+b)/2;
    end
    root = c
```

```
root = -1.4141
```

```
% Newton's Method
% function definition
    f = @(x)x-0.8-0.2*cos(x);
    df = @(x)1+0.2*sin(x);
    eps = 10^(-4);
    x_old = 1;
    x_new = x_old;
    for it = 1:1000
        x_new = x_old - f(x_old)/df(x_old);
        err = abs(x_new-x_old)
        if err<=eps
            break;
        end
        x_old=x_new;
    end
    x_new
```

```
root = -0.8655
```

```
% Secant method
% function definition
    f = @(x)-x^3-cos(x);
    df = @(x_new, x_old)(f(x_new)-f(x_old))/(x_new-x_old);
% initial conditions
    eps = 10^(-4);
    x_old = -2;
    x_new = 1;

    for it = 1:1000
        temp = x_new;
        x_new = x_new - f(x_new)/df(x_new, x_old)
```

```
        err = abs(x_new-x_old);
        if err<=eps
            break;
        end
        x_old = temp;
    end
    x_new
```

```
err = 0.0787
err = 3.0027e-04
err = 4.7046e-09
x_new = 0.9210
```

```
% One point iteration method
% function definition
    f = @(x) 3 - 2*log(1+exp(-x));
    b = zeros(20,1);
    eps = 10^(-6);
% Initial guess
    b(1) = 0;
    count = 1;

    for it = 1:1000
        b(it+1) = f(b(it));
        count = count+ 1;
        if abs(b(it) - b(it+1)) < eps
            break
        end
    end
    b(count)
```

```
ans = 2.8920
```

```
% Newton's Method for system of nonlinear equations
% F1 is an inverse of Jacobian of F
    F1 = @(x,y) [1/x, 1/x; 1/y, -1/y]/4;
    F = @(x,y) [(x^2 + y^2 - 4); (x^2 - y^2 - 1)];
    G = @(x,y) [x;y] - F1(x,y)*F(x,y);

    eps = 10^(-2);

    v0 = [1.6; 1.2];
    v1 = [0; 0];
    for it = 1:100
        v1 = G(v0(1), v0(2));
        if abs(v1 - v0) < eps
            it
            break
        end
        v0 = v1;
    end
```

```
it = 2
```

```
v1
```

```
v1 = 2×1
    1.5811
    1.2247
```

```

```

```
% Interpolation
    x_val = [1 1.25 1.6];
    y =@(x)(x-1).^(1/3);
    syms f(x);
    f(x) = interpolate(x_val, y(x_val))

    function f=interpolate(x, y)
        func = 0;
        for it=1:length(x)
            mask = (x~=x(it));
            func = func + generate_coef(x(it), x(mask))* y(it);
        end
        f = func;
    end

    function f = generate_coef(x_0, v)
        temp = 1;
        syms x;
        for it=1:length(v)
            temp = temp *(x-v(it))/(x_0-v(it));
        end
        f = temp;
    end
```