1. Solve by the following code

```
format long
% for h = 0.1
h1 = 0.1;
y0 = 0;
for i = 1:h1:2-h1
    y0 = y0 + h1 * f(y0,i);
end
disp("using h = 0.1, ans: ")
disp(y0)

% for h = 0.05
h2 = 0.05;
y1 = 0;
for i = 1:h2:2-h2
    y1 = y1 + h2 * f(y1,i);
end
disp("using h = 0.05, ans: ")
disp(y1)

% find the real value
myf = @(t, y) y^2 + t^2;
y0 = 0;
tspan = [1, 2];
[t, y] = ode45(myf, tspan, y0);
y_val = interp1(t, y, 2);
fprintf("real value: %.8f\n", y_val)
fprintf("error for h = 0.05 is %.8f\n", y_val - y1)

function y = f(y1, t)
    y = y1^2 + t^2;
end
```

Result :

```
>> Q1
using h = 0.1, ans:
   3.679686158559904

using h = 0.05, ans:
   4.558167628419639

real value: 6.70369045
error for h = 0.05 is 2.14552282
```

2. Use method of undetermined coefficient to find the coefficients

(2)

$$\int_0^h f(t)\,dt = C_0 f_{n-2} + C_1 f_{n-1} + C_2 f_n + C_3 f_{n+1}$$

$$f(t)=1 \rightarrow \int_0^h f(t)\,dt = h = C_0 + C_1 + C_2 + C_3$$

$$f(t)=t \rightarrow \int_0^h f(t)\,dt = \frac{h^2}{2} = C_0(-2h) + C_1(-h) + C_2(0) + C_3 \cdot h$$

$$f(t)=t^2 \rightarrow \int_0^h f(t)\,dt = \frac{h^3}{3} = C_0(-2h)^2 + C_1(-h)^2 + C_2(0) + C_3 h^2$$

$$f(t)=t^3 \rightarrow \int_0^h f(t)\,dt = \frac{h^4}{4} = C_0(-2h)^3 + C_1(-h)^3 + C_2(0) + C_3 h^3$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -2h & -h & 0 & h \\ 4h^2 & h^2 & 0 & h^2 \\ -8h^3 & -h^3 & 0 & h^3 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} h \\ \frac{h^2}{2} \\ \frac{h^3}{3} \\ \frac{h^4}{4} \end{bmatrix}$$

by the equation $\tilde{x}_{n+1} - x_n = \overset{C_0}{\left(\frac{1}{24}h\right)} f_{n-2} + \overset{C_1}{\left(-\frac{5}{24}h\right)} f_{n-1} + \overset{C_2}{\left(\frac{19}{24}h\right)} f_n + \overset{C_3}{\left(\frac{9}{24}h\right)} f_{n+1}$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -2h & -h & 0 & h \\ 4h^2 & h^2 & 0 & h^2 \\ -8h^3 & -h^3 & 0 & h^3 \end{bmatrix} \begin{bmatrix} \frac{h}{24} \\ -\frac{5}{24}h \\ \frac{19}{24}h \\ \frac{9}{24}h \end{bmatrix} = \begin{bmatrix} h \\ \frac{h^2}{2} \\ \frac{h^3}{3} \\ \frac{h^4}{4} \end{bmatrix}$$

$\Rightarrow$ the equation matches $\Rightarrow$ the coefficients are proved.

3.

(a) Derive and set the inputs to ode45()

(a)

$$y''' = t + 2y - ty'$$

$$x_1 = y$$

$$x_2 = y'$$

$$x_3 = y''$$

$$x_4 = y''' = t + 2y - ty'$$

$$\Rightarrow \begin{bmatrix} y' \\ y'' \\ y''' \end{bmatrix} = \begin{bmatrix} x_1' \\ x_2' \\ x_3' \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ t + 2x_1 - tx_2 \end{bmatrix}$$

Code :

```
format long

y0 = [0 1 0];
tspan = [0 0.2 0.4 0.6 0.8 1];

% solve the equation by sending the correct input into ode45()
[t, y] = ode45(@f, tspan, y0);

fprintf("(a)\n")
for i=1:5
    y_val = interp1(t, y, i*0.2);
    fprintf("    t = %.1f : y(%.1f) = %f, y'(%.1f) = %f, y''(%.1f) = %f\n",i*0.2,i*0.2,y_val(1), i*0.2, y_val(2), i*0.2, y_val(3) )
end

adams = zeros(6,3);
adams(1:4, 1:3) = y(1:4,1:3);

y_three = zeros(6,1);
for i = 1:4
    y_three(i) = t(i) + 2 * y(i,1) - t(i) * y(i, 2);
end
```

```
function dx = f(t,x)
    dx = [x(2) x(3) t+2*x(1)-t*x(2)]';
end
```

(b) (c) predictor and corrector derivation :

(b)

$$y_{n+1} = y_n + \frac{h}{24}\left(55 f_n - 59 f_{n-1} + 37 f_{n-2} - 9 f_{n-3}\right)$$

$$y_0 = y(0) = 0 \ , \ y_1 = y(0.2) \ , \ y_2 =$$

predictor

$$\begin{bmatrix} y_4 \\ y'_4 \\ y''_4 \end{bmatrix} = \begin{bmatrix} y_3 \\ y'_3 \\ y''_3 \end{bmatrix} + \frac{0.2}{24}\left(55\begin{bmatrix} x_2^{(3)} \\ x_3^{(3)} \\ 0.6+2x_1^{(3)}-0.6x_2^{(3)} \end{bmatrix} - 59\begin{bmatrix} x_2^{(2)} \\ x_3^{(2)} \\ 0.6+2x_1^{(2)}-0.6x_2^{(2)} \end{bmatrix} + 37\begin{bmatrix} x_2^{(1)} \\ x_3^{(1)} \\ 0.6+2x_1^{(1)}-0.6x_2^{(1)} \end{bmatrix} - 9\begin{bmatrix} x_2^{(0)} \\ x_3^{(0)} \\ 0.6+2x_1^{(0)}-0.6x_2^{(0)} \end{bmatrix}\right)$$

corrector

$$\begin{bmatrix} \tilde{y}_4 \\ \tilde{y}'_4 \\ \tilde{y}''_4 \end{bmatrix} = \begin{bmatrix} y_3 \\ y'_3 \\ y''_3 \end{bmatrix} + \frac{0.2}{24}\left(9\begin{bmatrix} x_2^{(4)} \\ x_3^{(4)} \\ 0.6+2x_1^{(4)}-0.6x_2^{(4)} \end{bmatrix} - 19\begin{bmatrix} x_2^{(3)} \\ x_3^{(4)} \\ 0.6+2x_1^{(3)}-0.6x_2^{(3)} \end{bmatrix} + (-5)\begin{bmatrix} x_2^{(2)} \\ x_3^{(2)} \\ 0.6+2x_1^{(2)}-0.6x_2^{(2)} \end{bmatrix} + \begin{bmatrix} x_2^{(1)} \\ x_3^{(1)} \\ 0.6+2x_1^{(1)}-0.6x_2^{(1)} \end{bmatrix}\right)$$

Code:

```
% for t = 0.8
% predictor
temp1 = zeros(1,3);
temp1 = adams(4,1:3) + 0.2/24 * (55*[adams(4,2) adams(4,3), y_three(4)] - 59*[adams(3,2) adams(3,3) y_three(3)] + 37*[adams(2,2) adams(2,3) y_three(2)] ...
    - 9*[adams(1,2) adams(1,3) y_three(1)]);
y_three(5) = t(5) + 2 * temp1(1,1) - t(5) * temp1(1,2);

% corrector
adams(5,1:3) = adams(4,1:3) + 0.2/24 * (9*[temp1(1,2) temp1(1,3) y_three(5)] + 19*[adams(4,2) adams(4,3), y_three(4)] - 5*[adams(3,2) adams(3,3) y_three(3)] + ...
    [adams(2,2) adams(2,3) y_three(2)]);
y_three(5) = t(5) + 2 * adams(5,1) - t(5) * adams(5,2);

% for t = 1
% predictor
temp1 = adams(5,1:3) + 0.2/24 * (55*[adams(5,2) adams(5,3), y_three(5)] - 59*[adams(4,2) adams(4,3) y_three(4)] + 37*[adams(3,2) adams(3,3) y_three(3)] ...
    - 9*[adams(2,2) adams(2,3) y_three(2)]);
y_three(6) = t(6) + 2 * temp1(1,1) - t(6) * temp1(1,2);
% corrector
adams(6,1:3) = adams(5,1:3) + 0.2/24 * (9*[temp1(1,2) temp1(1,3) y_three(6)]+ 19*[adams(5,2) adams(5,3), y_three(5)] - 5*[adams(4,2) adams(4,3) y_three(4)] + ...
    [adams(3,2) adams(3,3) y_three(3)]);
y_three(6) = t(6) + 2 * adams(6,1) - t(5) * adams(6,2);

fprintf("(b)\n")
fprintf("    when t = 1 and using Adams-Moulton method, y = %.10f\n", adams(6,1))

fprintf("(c)\n")
fprintf("    from the ode45() in (a), we know that when t = 1, y = %.10f, therefore the error is %.10f\n", y_val(1), abs(y_val(1)-adams(6,1)))
```

Ans :

```
>> Q3
(a)
    t = 0.2 : y(0.2) = 0.200133, y'(0.2) = 1.002666, y''(0.2) = 0.039989
    t = 0.4 : y(0.4) = 0.402132, y'(0.4) = 1.021311, y''(0.4) = 0.159659
    t = 0.6 : y(0.6) = 0.610778, y'(0.6) = 1.071741, y''(0.6) = 0.357416
    t = 0.8 : y(0.8) = 0.833967, y'(0.8) = 1.169218, y''(0.8) = 0.629160
    t = 1.0 : y(1.0) = 1.082545, y'(1.0) = 1.327833, y''(1.0) = 0.967159
(b)
    when t = 1 and using Adams-Moulton method, y = 1.0825646514
(c)
    from the ode45() in (a), we know that when t = 1, y = 1.0825451736, therefore the error is 0.0000194778
```

4.

Code :

Since the ode45() can not specify step size, so I use runge-kutta method to estimate in part (c)

```
A = [1 0 0 0 0;1 -2+(pi/4)^2/4 1 0 0; 0 1 -2+(pi/4)^2/4 1 0 ;0 0 1 -2+(pi/4)^2/4 1; 0 0 0 0 1];
b = [0 0 0 0 2]';
x = A\b;
fprintf("(a)\n    y(theta) = \n")
disp(x)

syms theta;
y = 2*sin(theta/2);
theta_values = [pi/4, pi/2, 3*pi/4];
ans1 = double(subs(y, theta, theta_values));
fprintf("                       real value      estimated value       error\n")
for i = 1:3
    fprintf("    y(%.1d * pi / 4) :      %.8f        %.8f       %.8f%%\n", i, ans1(i), x(i+1), abs(ans1(i)-x(i+1))/ans1(i))
end


fprintf("\n(b)\n")
% choose h = pi/30
h = pi/30;
A_new = zeros(31,31);
for i = 2:30
    A_new(i,i-1) = 1;
    A_new(i,i) = -2+h^2/4;
    A_new(i,i+1) = 1;
end
A_new(31,31) = 1;
A_new(1,1) = 1;
b_new = zeros(31,1);
b_new(31) = 2;

% calculate Ax = b
x_new = A_new\b_new;


% substitute each theta into y'
theta_values_new = zeros(29,1);
for i=1:29
    theta_values_new(i) = i*pi/30;
end
ans_new = double(subs(y, theta, theta_values_new));

fprintf("                        real value      estimated value       error\n")
for i = 1:29
    fprintf("    y(pi * %2d / 30) :    %.8f        %.8f       %.8f%%\n", i, ans_new(i), x_new(i+1), abs(ans_new(i)-x_new(i+1))/ans_new(i))
end
% find all the errors
errors = abs(ans_new - x_new(2:30,1))./ans_new;

% display the maximum
fprintf("    maximum error = %.8f\n", max(errors))


fprintf("(c)\n")
fprintf("    use secant method with runge-kutta method with specific h\n")
fprintf("    using h = pi/2 can reduce the error small enough\n\n")
h = pi/2;                  % the h (can be changed directly)
U3 = secant(@shooting, [0, 0.7], [0, 1.2], 1e-5, h);
[dummy, x, Ux1] = shooting([0, 0.7], h);
[dummy, x, Ux2] = shooting([0, 1.2],h);
[dummy, x, Ux3] = shooting([0, U3(2)], h);
plot(x, Ux1(:,1), x, Ux2(:,1), x, Ux3(:,1));

theta_values_new = zeros(pi/h-1,1);
for i=1:pi/h-1
    theta_values_new(i) = i*h;
end
ans_new = double(subs(y, theta, theta_values_new));

errors = abs(ans_new - Ux3(2:pi/h, 1))./ans_new;

fprintf("                        real value      estimated value\n")
for i = 1:pi/h - 1
    fprintf("    y(pi * %d / 2) :      %.8f        %.8f\n", i, ans_new(i), Ux3(i+1))
end
fprintf("    maximum error = %.8f\n", max(errors))

function [P, x, U] = shooting(U0, h)
    [x, U] = rk(h, U0(2));
    P = U(length(x),1) - 2;
end
```

```
function x2 = secant(f, x0, x1, tol, h)
    if abs(f(x0, h)) < abs(f(x1, h))
        tmp = x0;
        x0 = x1;
        x1 = tmp;
    end
    x2 = x1 - f(x1, h)*(x0-x1)/(f(x0, h)-f(x1, h));
    while abs(f(x2, h)) > tol
        x0 = x1;
        x1 = x2;
        x2 = x1 - f(x1, h)*(x0-x1)/(f(x0, h)-f(x1, h));
    end
end

function [x, ans1] = rk(h, first)
    dy_dx = @(x, y, z) z;
    dz_dx = @(x, y, z) -y/4;

    % Define the step size and the number of steps
    num_steps = pi/h;

    % Initialize arrays to store the x, y, and z values
    x = zeros(num_steps+1, 1);
    y = zeros(num_steps+1, 1);
    z = zeros(num_steps+1, 1);

    % Set the initial values
    x(1) = 0;
    y(1) = 0;
    z(1) = first;

    % Runge-Kutta method
    for i = 1:num_steps
        k1y = h * dy_dx(x(i), y(i), z(i));
        k1z = h * dz_dx(x(i), y(i), z(i));

        k2y = h * dy_dx(x(i) + h/2, y(i) + k1y/2, z(i) + k1z/2);
        k2z = h * dz_dx(x(i) + h/2, y(i) + k1y/2, z(i) + k1z/2);

        k3y = h * dy_dx(x(i) + h/2, y(i) + k2y/2, z(i) + k2z/2);
        k3z = h * dz_dx(x(i) + h/2, y(i) + k2y/2, z(i) + k2z/2);

        k4y = h * dy_dx(x(i) + h, y(i) + k3y, z(i) + k3z);
        k4z = h * dz_dx(x(i) + h, y(i) + k3y, z(i) + k3z);

        x(i+1) = x(i) + h;
        y(i+1) = y(i) + (k1y + 2*k2y + 2*k3y + k4y)/6;
        z(i+1) = z(i) + (k1z + 2*k2z + 2*k3z + k4z)/6;
    end

    ans1 = [y z];

end
```

Ans:

```
>> Q4
(a)
   y(theta) =
                       0
   0.770150209564288
   1.421533576975207
   1.853698599892208
   2.000000000000000

                          real value      estimated value       error
   y(1 * pi / 4) :        0.76536686        0.77015021        0.00624974%
   y(2 * pi / 4) :        1.41421356        1.42153358        0.00517603%
   y(3 * pi / 4) :        1.84775907        1.85369860        0.00321445%
```

(b)

	real value	estimated value	error
y(pi * 1 / 30) :	0.10467191	0.10468386	0.00011418%
y(pi * 2 / 30) :	0.20905693	0.20908073	0.00011386%
y(pi * 3 / 30) :	0.31286893	0.31290439	0.00011334%
y(pi * 4 / 30) :	0.41582338	0.41587021	0.00011261%
y(pi * 5 / 30) :	0.51763809	0.51769589	0.00011166%
y(pi * 6 / 30) :	0.61803399	0.61810228	0.00011050%
y(pi * 7 / 30) :	0.71673590	0.71681411	0.00010912%
y(pi * 8 / 30) :	0.81347329	0.81356075	0.00010752%
y(pi * 9 / 30) :	0.90798100	0.90807697	0.00010570%
y(pi * 10 / 30) :	1.00000000	1.00010364	0.00010364%
y(pi * 11 / 30) :	1.08927807	1.08938848	0.00010136%
y(pi * 12 / 30) :	1.17557050	1.17568669	0.00009883%
y(pi * 13 / 30) :	1.25864078	1.25876169	0.00009606%
y(pi * 14 / 30) :	1.33826121	1.33838572	0.00009304%
y(pi * 15 / 30) :	1.41421356	1.41434050	0.00008976%
y(pi * 16 / 30) :	1.48628965	1.48641778	0.00008621%
y(pi * 17 / 30) :	1.55429192	1.55441996	0.00008237%
y(pi * 18 / 30) :	1.61803399	1.61816061	0.00007825%
y(pi * 19 / 30) :	1.67734114	1.67746498	0.00007383%
y(pi * 20 / 30) :	1.73205081	1.73217048	0.00006909%
y(pi * 21 / 30) :	1.78201305	1.78212714	0.00006403%
y(pi * 22 / 30) :	1.82709092	1.82719800	0.00005861%
y(pi * 23 / 30) :	1.86716085	1.86725949	0.00005283%
y(pi * 24 / 30) :	1.90211303	1.90220179	0.00004666%
y(pi * 25 / 30) :	1.93185165	1.93192909	0.00004008%
y(pi * 26 / 30) :	1.95629520	1.95635989	0.00003307%
y(pi * 27 / 30) :	1.97537668	1.97542723	0.00002559%
y(pi * 28 / 30) :	1.98904379	1.98907882	0.00001761%
y(pi * 29 / 30) :	1.99725907	1.99727723	0.00000909%

maximum error = 0.00011418

(c)

use secant method with runge-kutta method with specific h
using h = pi/2 can reduce the error small enough

	real value	estimated value
y(pi * 1 / 2) :	1.41421356	1.41356901

maximum error = 0.00045577

Derivation for (a)(b):

$$y'' = -\frac{y}{4}$$

$$x_{i-1} - 2x_i + x_{i+1} = h^2 f\left(t_i, x_i, \frac{x_{i-1} - x_{i+1}}{2}\right)$$

$$x_0 - 2x_1 + x_2 = h^2\left(-\frac{x_1}{4}\right), \quad h = \frac{\pi}{4}$$

$$x_0 - \frac{7}{4}x_1 + x_2 = 0$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -\frac{7}{4} & 1 & 0 & 0 \\ 0 & 1 & -\frac{7}{4} & 1 & 0 \\ 0 & 0 & 1 & -\frac{7}{4} & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 2 \end{bmatrix}$$

$$u_1 = y$$

$$u_2 = y'$$

$$\begin{bmatrix} u_1' \\ u_2' \end{bmatrix} = \begin{bmatrix} u_2 \\ u_2'' = -\frac{u_1}{4} \end{bmatrix}$$

5.

Derivation :

$h = \frac{1}{4}$ $\quad t_{-1} = -0.25,\ t_0 = 0$, subinterval $(0,1) = 4$

$$\begin{bmatrix} u_1' \\ u_2' \end{bmatrix} = \begin{bmatrix} u_2 \\ t^3 + tx' - t^2x \end{bmatrix} = \begin{bmatrix} u_2 \\ t^3 + tu_2 - t^2u_1 \end{bmatrix}$$

$$x_0 - 2x_1 + x_2 = h^2 f\left(t_1, x_1, \frac{x_2 - x_0}{2h}\right)$$

$$= h^2\left(t_1 \frac{x_2 - x_0}{2h} + t_1^3 - t_1^2 x_1\right)$$

$$x_0 - 2x_1 + x_2 = \frac{1}{16}\left(-2t_1x_0 - t_1^2x_1 + 2t_1x_2 + t_1^3\right)$$

$$\left(\frac{1}{8}t_1 + 1\right)x_0 + \left(\frac{1}{16}t_1^2 - 2\right)x_1 + \left(1 - \frac{1}{8}t_1\right)x_2 = \frac{t_1^3}{16}$$

1

$$\Rightarrow \left(\frac{1}{8}t_n + 1\right)x_{n-1} + \left(\frac{1}{16}t_n^2 - 2\right)x_n + \left(1 - \frac{1}{8}t_n\right)x_{n+1} = \frac{t_n^3}{16}$$

2

initial constraint : $x_0 + \frac{x_1 - x_{-1}}{\frac{1}{2}} - x_4 + \frac{x_5 - x_3}{\frac{1}{2}} = 3$

3

$x_0 = \frac{5}{2}$

1 2 3 $\rightarrow$ solve

Code :

```
t = -0.25:0.25:1.25;
A = zeros(7,7);

A(1,1:7) = [-2 1 2 0 -2 -1 2];
A(2,2) = 1;
b = zeros(6,1);
b(1) = 3;
b(2) = 5/2;
for i = 3:7
    A(i,i-2) = 1+t(i-1)/8;
    A(i,i-1) = -2+t(i-1)/16;
    A(i,i) = 1-t(i-1)/8;
    b(i) = t(i-1)^3 /16;
end
x = A\b;
fprintf("    subinterval h = 0.25\n")
fprintf("    x_values for t = 0 to t = 1 having h = 0.25 :\n")
disp(x(2:6))
```

Ans :

```
>> Q5
    subinterval h = 0.25
    x_values for t = 0 to t = 1 having h = 0.25 :
   2.500000000000000
   3.362484649399423
   4.227387910866560
   5.075032010167096
   5.864643922590135
```

v