

pb1	1) euler's method 함수(3), y값 계산(2)
	2) heun's method 함수(3), corrector 갱신(5), y값 계산(2)
	3) 겹쳐서 plot(1), x,y label(1), title(1), legend(1), line option(1), ppt 미 첨부(-1)
	4) 상대오차 계산(3), 최대 오차 계산(1), 5% 찾는 while or for문(5), 최종 step size 출력(1)
pb2	1) ODE 함수 정의(2), MATLAB 함수로 풀기(3), 선형보간으로 dT/dx 구하기(2), MATLAB 함수써서 T값 계산(3)
	2) 파라미터 값 정의(1), f vector(3), K matrix(4), T값 계산(2)
	3) 겹쳐서 plot(1), x,y label(1), title(1), legend(1), line option(1), ppt 미 첨부(-1)
	4) dT/dx 초기값 넣기(2), T 초기값 넣기(2), (dT/dx)^2 계산(3), 이론해 계산(2), 결과비교 scope(1), ppt 미 첨부(-1)
pb3	1) 모델링(3), parameter setting(2)
	2) case별 오차가 +/-10% 2개 이상 차이 시 개당 -1점(4), 결론(1)
pb4	PID(2), saturation(1), input inertia(3), gear(2), clutch(5), output inertia(3), 차속 계산(1), 차속에러(2), 공기저항(2), 구름저항(2), 총저항(1), 가속시간(1), ppt 미 첨부(-1)
timeover	제출시간 2분 초과 시 마다 -1점(6:19분 종료)

1. (1)~(3) refer to lecture

(4)

```

clc; clear all; close all;
%% analytic solution
step=0.2;
for i=1:200;
    step=step-0.001;
    t_a = [0:step:4]';
    y_a = -0.5*t_a.^4+4*t_a.^3-10*t_a.^2+8.5*t_a+1;

    %% numerical solution
    dydt = @(t,y) -2*t.^3+12*t.^2-20*t+8.5;
    [t_n,y_n] = eulode(dydt,[0,4],1,step);

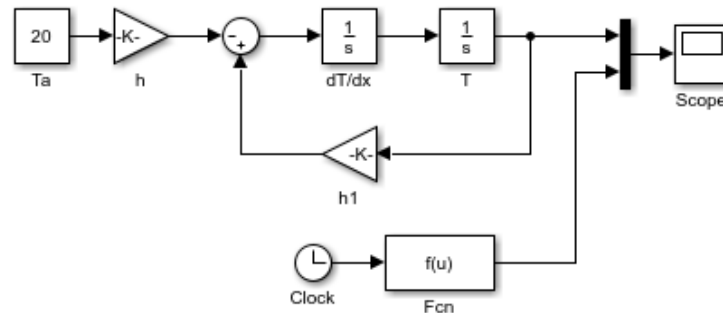
    %% error calculation
    err=max((abs(y_a-y_n)./y_a*100))
    if err<5
        break
    end
end
step

function [t_n,y_n] = eulode(dydt,tspan,y0,h)
ti = tspan(1);tf = tspan(2);
if ~(tf>ti)
    error('upper limit must be greater than lower')
end
t_n = (ti:h:tf)'; n = length(t_n);
y_n = y0*ones(n,1); % preallocate y to improve efficiency
for i = 1:n-1 % implement Euler's method
    y_n(i+1) = y_n(i) + dydt(t_n(i),y_n(i))*h;
end
end

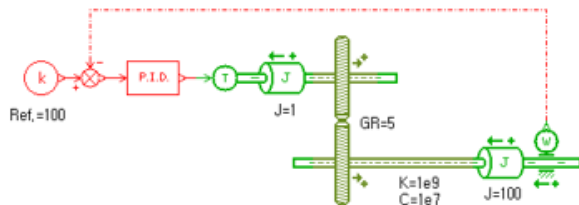
```

2. (1)~(3) refer to lecture

(4)



3.



error	Kp=1	Kp=5	Kp=10
Ki=0.1	1.17	0.97	0.25
Ki=1	40.21	4.99	1.70
Ki=5	29.74	1.01	1.16

min.
error

4.

