```
%% Name: Mohammed Al-Sayegh, ECE 414 Homework - PID Tuner

%% Part 1, P control
clc
close all

% Using pidtuner to generate a PI controller
C_p = pidtune(G, 'P');

% Pass the default terms of Kp and Ki baseline tune of PI by pidTuner
Tune = pidTuner(G, C_p);
waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTuner');
disp('Kp = ');
disp(C_p.Kp);
disp('The step info of P controller using stepinfo:');
disp(stepinfo(C_p));
disp('The step info of P controller using getallspecs:');
disp(getallspecs(G,C_p));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_p = (C_p*G)/(1+(G*C_p));

%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;

step(T_p)
grid on;

legend('Baseline');
title('System Step Response of PI Contoller');
hold off;

%% Plot the Baseline vs. tuned system Contoller Effort Step Response
figure(2)
hold on;

Tu = T_p/G;
step(Tu)

grid on;

legend('Baseline');
title('Contoller Effort Step Response of P');
hold off;

%% Part 2, PD control
```

```
clc
close all

% Using pidtuner to generate a PD controller
C_pd = pidtune(G, 'PD');

% Pass the default terms of Kp and Ki baseline tune of PI by pidTuner
Tune = pidTuner(G, C_pd);
waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTuner');
disp('Kp = ');
disp(C_pd.Kp);
disp('Kd = ');
disp(C_pd.Kd);
disp('Cannot simulate the time response of improper (non-causal) PD contorller\n');
disp('The step info of PD controller using getallspecs:');
disp(getallspecs(G,C_pd));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_pd = (C_pd*G)/(1+(G*C_pd));

%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;

step(T_pd)
grid on;

legend('Baseline');
title('System Step Response of PD Contoller');
hold off;

%% Plot the Baseline vs. tuned system Contoller Effort Step Response
figure(2)
hold on;

Tu = T_pd/G;
% step(Tu)

grid on;

legend('Baseline', 'Tuned');
title('Contoller Effort Step Response of PD');
hold off;

%% Part 3, PI control
clc
close all
```

```matlab
% Using pidtuner to generate a PI controller
C_pi = pidtune(G, 'PI');

% Pass the default terms of Kp and Ki baseline tune of PI by pidTuner
Tune = pidTuner(G, C_pi);
waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTuner');
disp('Kp = ');
disp(C_pi.Kp);
disp('Ki = ');
disp(C_pi.Ki);
disp('The step info of PI controller using stepinfo:');
disp(stepinfo(C_pi));
disp('The step info of PI controller using getallspecs:');
disp(getallspecs(G,C_pi));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_pi = (C_pi*G)/(1+(G*C_pi));

%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;
subplot(1,2,1);
step(T_pi)
grid on;

legend('Baseline');
title('System Step Response of PI Contoller');

% Plot the Baseline vs. tuned system Contoller Effort Step Response

hold on;
subplot(1,2,2)
Tu = T_pi/G;
step(Tu)
grid on;

legend('Baseline');
title('Contoller Effort Step Response of PI');
hold off;

%% Part 4, PID control
pause(1);
close all;

%% Using pidtuner to generate a PID controller
C_pid = pidtune(G, 'PID');
```

```matlab
% Pass the default terms of Kp and Ki baseline tune of PI by pidTuner
Tune = pidTuner(G, C_pid);
waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTune');
disp('Kp = ');
disp(C_pid.Kp);
disp('Ki = ');
disp(C_pid.Ki);
disp('Kd = ');
disp(C_pid.Kd);
%disp('The step info of PID controller using stepinfo:');
%disp(stepinfo(C_pid));
disp('The step info of PID controller using getallspecs:');
disp(getallspecs(G,C_pid));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_pid = (C_pid*G)/(1+(G*C_pid));


%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;

step(T_pid)
grid on;

legend('Baseline');
title('System Step Response of PID Contoller');
hold off;

%% Plot the Baseline vs. tuned system Contoller Effort Step Response
figure(2)
hold on;

Tu = T_pid/G;
%step(Tu)
grid on;

legend('Baseline');
title('Contoller Effort Step Response of PID');
hold off;

%% Part 5, PIDF control
clc;
pause(1);
close all;
```

```matlab
%% Using pidtuner to generate a PIDF controller
C_pidf = pidtune(G, 'PIDF');

% Pass the default terms of Kp, Ki and Kd baseline tune of PID by pidTuner
Tune = pidTuner(G, C_pidf);
waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTune');
disp('Kp = ');
disp(C_pidf.Kp);
disp('Ki = ');
disp(C_pidf.Ki);
disp('Kd = ');
disp(C_pidf.Kd);
disp('The step info of PIDF controller using stepinfo:');
disp(stepinfo(C_pidf));
disp('The step info of PIDF controller using getallspecs:');
disp(getallspecs(G,C_pidf));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_pidf = (C_pidf*G)/(1+(G*C_pidf));

%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;

subplot(1,2,1)
step(T_pidf)
grid on;

legend('Baseline');
title('System Step Response of PIDF Contoller');

% Plot the Baseline vs. tuned system Contoller Effort Step Response

subplot(1,2,2)
Tu = T_pidf/G;
step(Tu)
grid on;

legend('Baseline');
title('Contoller Effort Step Response of PIDF');
hold off;

%% Part 6, PDF control
clc;
pause(1);
close all;
```

```
%% Using pidtuner to generate a PDF controller
C_pdf = pidtune(G, 'PDF');

% Pass the default terms of Kp, Ki and Kd baseline tune of PID by pidTuner
%Tune = pidTuner(G, C_pdf);
%waitfor(Tune);

% Display base terms that given by pidTuner
disp('Parameters from pidTune');
disp('Kp = ');
disp(C_pdf.Kp);
disp('Ki = ');
disp(C_pdf.Ki);
disp('Kd = ');
disp(C_pdf.Kd);
disp('The step info of PIDF controller using stepinfo:');
disp(stepinfo(C_pdf));
disp('The step info of PIDF controller using getallspecs:');
disp(getallspecs(G,C_pdf));

% Baseline controller effort and system transfer functions
% Baseline System Transfer Function
T_pdf = (C_pdf*G)/(1+(G*C_pdf));

%% Plot the Baseline vs. tuned system Step Response
figure(1);
hold on;

subplot(1,2,1)
step(T_pdf)
grid on;

legend('Baseline');
title('System Step Response of PDF Contoller');
hold off;

%% Plot the Baseline vs. tuned system Contoller Effort Step Response
figure(2)
hold on;
subplot(1,2,2)
Tu = T_pdf/G;
step(Tu)
grid on;

legend('Baseline');
title('Contoller Effort Step Response of PDF');
hold off;
```