```matlab
% lab1 - 334-384
% =========================================================================

clear
close all                % close all figures

%data = load('data_lab1_green_2022.txt','-ascii');      % using "green" box

%data(:,2) = data(:,2) + 12;    % for green box, hours are PM; convert them to 24-hour format

data = load('data_lab1_grey_2022.txt','-ascii');        % using "grey" box

indbase = find(data(:,1)==0);   % lines in 'data' with base-station readings
indfield = find(data(:,1)~=0);  % lines with all other (field-station) readings

% step #1
% convert time into hours from 3pm and create new column #8

data(:,8) = (data(:,2)-15) + data(:,3)/60;

% step #2 Calibrate dial readings
% put calibrated gravity values in new column #9

data(:,9) = (data(:,7)-4500)*1.04848 + 4712.62;

        % plot all raw dial and calibrated readings

figure(2)
hold on

plot(data(indbase,8),data(indbase,7),'ok')      % dial readings: base station black, others red
plot(data(indfield,8),data(indfield,7),'or')

plot(data(indbase,8),data(indbase,9),'xk')      % calibrated values: base station black, others
red
plot(data(indfield,8),data(indfield,9),'xr')

xlim([0.2,3.5])                 % set wide time range so that the legend does not cover data
points

xlabel('time (hrs)')
ylabel('Gravity (mGal)')
title('Data')
legend('Dial Base','Dial Field','Calibrated Base','Calibrated Field')

% step #3  Perform drift correction

figure(3)
hold on

p = polyfit(data(indbase,8),data(indbase,9),1); % linear fitting for base_gravity(time)

drift_model = polyval(p,data(:,8));

plot(data(indbase,8),data(indbase,9),'ok')      % plot base readings vs time

plot(data(:,8),drift_model,'-b')                % drift model line is supposed to be blue

        % take reference level at the second base station reading

reflevel = data(indbase(2),9);

plot([data(1,8),data(end,8)],[ reflevel, reflevel],'r-')  % drawing reference level by red line

        % Reference_level_drift_model

drift = drift_model - reflevel;

        % put drift-corrected data in column #10

data(:,10) = data(:,9) - drift;
```

```
xlim([0.2,3.5])                 % set wide time range so that the legend does not cover data
points

xlabel('time (hrs)')
ylabel('Gravity (mGal)')
title('Calibration and drift')
legend('Base station','Drift model','Reference')

figure(31)
subplot(3,1,1)
hold on

plot(data(indfield,8),data(indfield,10),'ok','markersize',6,'markerfacecolor','k')
plot(data(indbase,8),data(indbase,10),'ob','markersize',6,'markerfacecolor','b')

plot([data(1,8),data(end,8)],[ reflevel, reflevel],'r-')  % drawing reference level by red line

xlim([0.2,3.5])                 % set wide time range so that the legend does not cover data
points
xlabel('time (hrs)')
ylabel('Gravity (mGal)')
title('Drift-corrected gravity')
legend('field stations','base','reference level')

% #step #4  Measure the repeatability (random error) in the data

sumrep = 0.0;
nrep = 0;

        % try all possible station number and see if we have repeated
        % readings at these stations

for i= 0:size(data,1)
        ind = find(data(:,1)==i);       % rows of all records with station number i

        if length(ind) > 1              % more than one station - have some repetitions
                r = data(ind,10);       % corrected gravity records from all repeated readings
at this station
                sumrep = sumrep + sum((r(2:end,1)-r(1:(end-1),1)).^2);
                                        % sum of squared differences of successive readings

                nrep = nrep + length(ind) - 1;  % count the number of repeated pairs
        end
end

repeat = sqrt(sumrep/nrep)      % print out repeatability value (square root of mean square)

% step #5 Put your field data on an absolute scale.

data(:,11) = data(:,10) + 981121.10 - reflevel;         % calibrated absolute gravity values
column

%step #6 Evaluate the latitude correction

cos2 = cosd(37.8)^2;    % squared cosine in the formula
g_lat = 978032.67714*((1+0.00193185138639*cos2)/sqrt(1-0.00669437999013*cos2));

data(:,12) = g_lat;             % put a column of g_lat in data table
data(:,13) = data(:,11) - g_lat;  % latitude-corrected gravity in column 13

% Measure slope of the free-air effect. It should be negative and abs. value less than 0.3086

slope = polyfit(data(:,6),data(:,13),1)
slope_model = polyval(slope,data(:,6));

%step#7 free-air correction
data(:,14) = 0.3086*data(:,6)+data(:,13);


subplot(3,1,2)
hold on
plot(data(:,6),slope_model,'-b')
```

```
plot(data(:,6),data(:,13),'ob','markersize',6,'markerfacecolor','b')

xlabel('Height(m)')
ylabel('Gravity (mGal)')
title('Latitude-corrected gravity')

subplot(3,1,3)

plot(data(:,6),data(:,14),'ob','markersize',6,'markerfacecolor','g')

xlabel('Height(m)')
ylabel('Gravity (mGal)')
title('Free-air corrected gravity')
```