Name	Team Number
Tomoki Koike	**R06**

swd

AAE 251: Introduction to Aerospace Design

Assignment 9— Aircraft Performance II

Due Tuesday April 9, 10:00 am on Blackboard

Instructions

Write or type your answers into the appropriate boxes. ***Make sure you submit a single PDF on Blackboard.***

Make sure you keep a record of submission receipts or the confirmation emails after each submission as a proof that your submission was accepted.

	Score	Max
Question 1		**16**
Question 2		**6**
Question 3		**14**
Question 4		**6**
Question 5		**7**
Question 6		**8**
Question 7		**8**
TOTAL		**65**

MATLAB FUNCTIONS

>> Function: Atmospheric Density Calculator

Description: This function computes the air density for each corresponding altitude

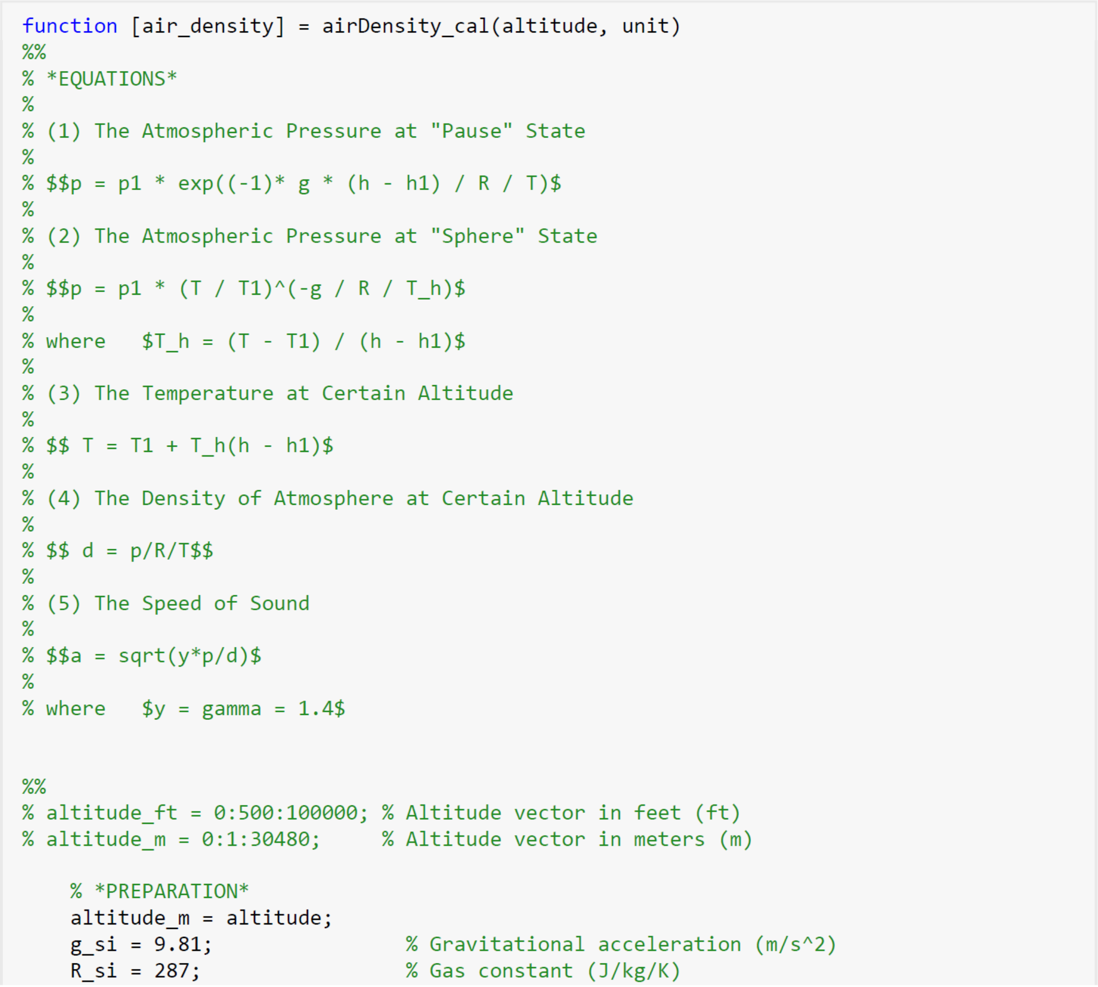
This function is from HW #1

(1) INPUTS:

- altitude: altitude in the atmosphere [m] or [ft]
- unit: input to decide whether to do the computations in SI or English units

(2) OUTPUTS:

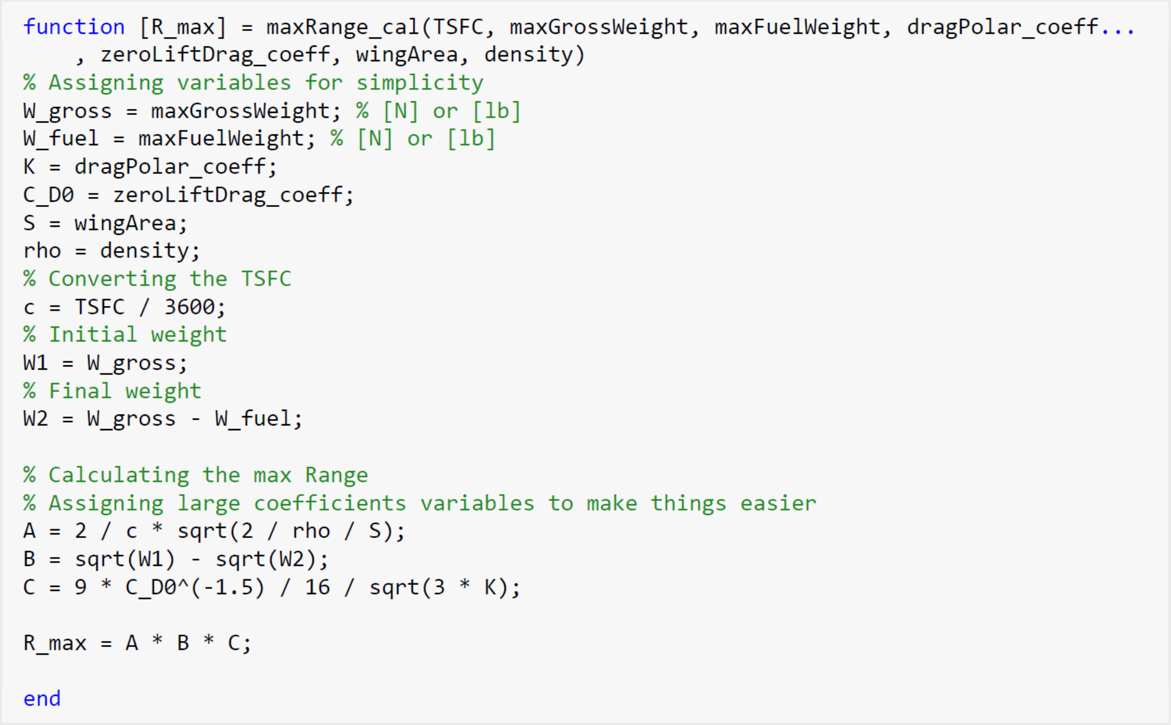
- air density: the air density [kg/m^3] or [slug/ft^3]



```
function [air_density] = airDensity_cal(altitude, unit)
%%
% *EQUATIONS*
%
% (1) The Atmospheric Pressure at "Pause" State
%
% $$p = p1 * exp((-1)* g * (h - h1) / R / T)$
%
% (2) The Atmospheric Pressure at "Sphere" State
%
% $$p = p1 * (T / T1)^(-g / R / T_h)$
%
% where   $T_h = (T - T1) / (h - h1)$
%
% (3) The Temperature at Certain Altitude
%
% $$ T = T1 + T_h(h - h1)$
%
% (4) The Density of Atmosphere at Certain Altitude
%
% $$ d = p/R/T$$
%
% (5) The Speed of Sound
%
% $$a = sqrt(y*p/d)$
%
% where   $y = gamma = 1.4$


%%
% altitude_ft = 0:500:100000; % Altitude vector in feet (ft)
% altitude_m = 0:1:30480;     % Altitude vector in meters (m)

    % *PREPARATION*
    altitude_m = altitude;
    g_si = 9.81;                 % Gravitational acceleration (m/s^2)
    R_si = 287;                  % Gas constant (J/kg/K)
    gamma = 1.4;                % Adiabatic Index or Isentropic Expansion Constant
    lapse_rate_m = [-6.5*power(10,-3), 3*power(10,-3), -4.5*power(10,-3), ...
    4*power(10,-3)];          % Temperature lapse rates (K/m)
    mark_height_m = [0, 11, 25, 47, 53, 79, 90, 105]*1000;
                              % Height at which the the state changes
                              % from "pause" to "sphere" or vice versa
                              % (m)
    initial_temp_m = [288.16, 216.66, 282.66, 165.66, 256.66];
                              % Initial temperatures (K) where the
                              % state changes from "pause" to "sphere"
                              % or vice versa
    % *Temperature*
    %
    % Finding the temperature by altitude (K)
    % Meters
    temp_m = tempCal(initial_temp_m, altitude_m, mark_height_m,lapse_rate_m);
    % *Pressure*
    pressure_m = pressureCal(g_si, R_si, temp_m, initial_temp_m,...
        altitude_m, mark_height_m, lapse_rate_m, "SI");
    % *Density*
    density_m = pressure_m ./ temp_m / R_si;

    % *PREPARATION*
    altitude_ft = altitude;
    g_eng = 32.174;             % Gravitational acceleration (ft/s^2)
    R_eng = 1716.27;            % Gas constant (ft^2/s^2R)
    gamma = 1.4;                % Adiabatic Index or Isentropic Expansion Constant
    lapse_rate_ft = lapse_rate_m / 3.28084 * 1.8;
                              % Temperature lapse rates (R/ft)
    mark_height_ft = mark_height_m * 3.28084;
                              % Height corresponding to mark_height_m
                              % in feet (ft)
    initial_temp_ft = [518.688, 389.988, 515.988, 298.188, 461.988];
                              % Initial temperatures (R) where the
                              % state changes from "pause" to "sphere"
                              % or vice versa
    % *Temperature*
    %
    % Finding the temperature by altitude (K)
    % Feet
    temp_ft = tempCal(initial_temp_ft, altitude_ft, mark_height_ft, lapse_rate_ft);
    % *Pressure*
    pressure_ft = pressureCal(g_eng, R_eng, temp_ft, initial_temp_ft,...
        altitude_ft, mark_height_ft, lapse_rate_ft, "ENG");
    % *Density*
    density_ft = pressure_ft ./ temp_ft / R_eng;
if unit == "SI"
    air_density = density_m;
else
    air_density = density_ft;
end
end
```

>> Max Range Calculator

-- this function will calculate the maximum range for an aircraft



```
function [R_max] = maxRange_cal(TSFC, maxGrossWeight, maxFuelWeight, dragPolar_coeff...
    , zeroLiftDrag_coeff, wingArea, density)
% Assigning variables for simplicity
W_gross = maxGrossWeight; % [N] or [lb]
W_fuel = maxFuelWeight; % [N] or [lb]
K = dragPolar_coeff;
C_D0 = zeroLiftDrag_coeff;
S = wingArea;
rho = density;
% Converting the TSFC
c = TSFC / 3600;
% Initial weight
W1 = W_gross;
% Final weight
W2 = W_gross - W_fuel;

% Calculating the max Range
% Assigning large coefficients variables to make things easier
A = 2 / c * sqrt(2 / rho / S);
B = sqrt(W1) - sqrt(W2);
C = 9 * C_D0^(-1.5) / 16 / sqrt(3 * K);

R_max = A * B * C;

end
```

>> Maximum Velocity Calculator for Max Power

-- this function allows to calculate the maximum velocity for the corresponding maximum power of an aircraft .

```
function [V_max] = maxVel_for_power(max_power, weight, density, wing_area, zeroLiftDrag_coeff,

% Setting simplified variables for the descriptive input variables
rho = density;
W = weight;
S = wing_area;
C_D0 = zeroLiftDrag_coeff;
K = dragPolar_coeff;
P = max_power;

% Defining the system to solve
syms V
% Lumping up large coefficients
A = 0.5 * rho * S * C_D0;
B = 2 * K * W^2 / rho / S;
% Setting the system equation
eqn = A * V^4 + B == P * V;
% Solving the Power equation for V
V_max_vec = solve(eqn, V);

% eliminating the complex values in the vector
V_max_vec = double(V_max_vec);
z = [];
for k = 1:length(V_max_vec)
    if imag(V_max_vec(k)) == 0
        z = [z, real(V_max_vec(k))];
    end
end

% Therefore the maximum Velocity becomes
V_max = max(z);

end
```

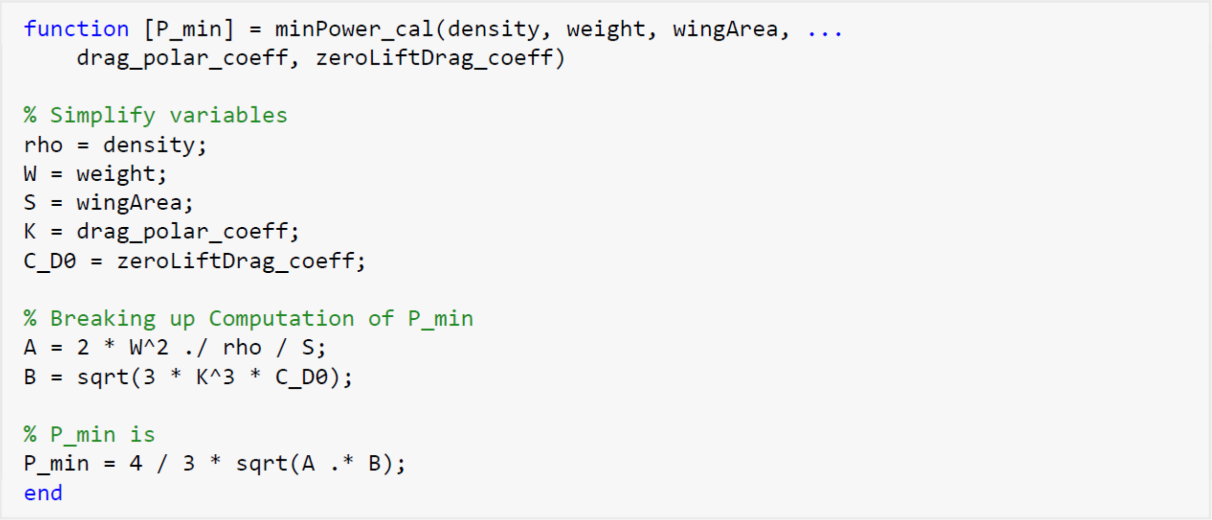
>> Maximum Velocity Calculator for Max Thrust

-- this function allows to calculate the maximum velocity for the corresponding maximum thrust of an aircraft .

```
function [V_max] = maxVel_for_thrust(thrust, weight, density, wing_area, zeroLiftDraf_coeff, d
% Assigning simplified variables to the input variables
T = thrust; % [N] or [lb]
W = weight; % [m] or [ft]
rho = density; % [kg/m^3] or [slug/ft^3]
S = wing_area; % [m^2] or [ft^2]
C_D0 = zeroLiftDraf_coeff;
K = dragPolar_coeff;

% Defining the system to solve
syms V
% Lumping up large coefficients
A = 0.5 * rho * S * C_D0;
B = 2 * K * W^2 / rho / S;
% Setting the system equation
eqn = A * V^4 + B == T * V^2;
% Solving the Thrust equation for V
V_max_vec = solve(eqn, V);

% eliminating the complex values in the vector
V_max_vec = double(V_max_vec);
z = [];
for k = 1:length(V_max_vec)
    if imag(V_max_vec(k)) == 0
        z = [z, real(V_max_vec(k))];
    end
end

% Therefore the maximum Velocity becomes
V_max = max(z);
end
```

>> Minimum Power Calculator

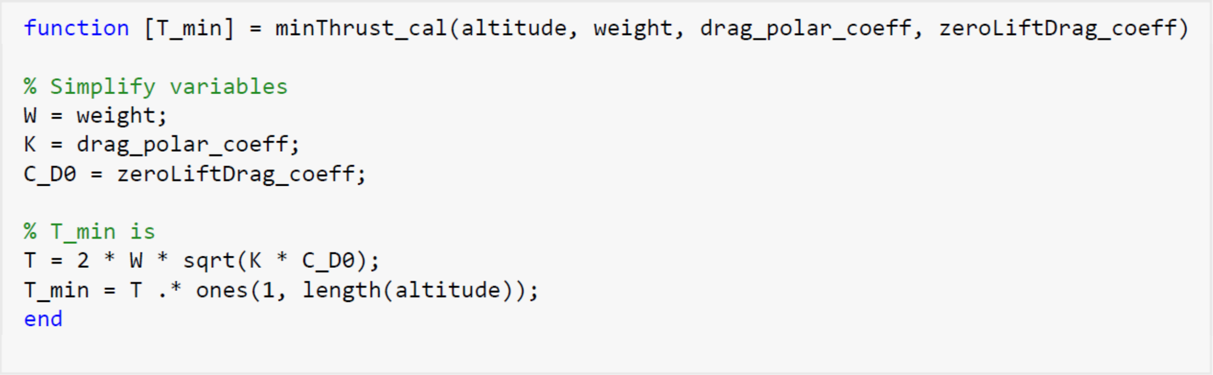
-- This function calculates the minimum power as a function of the altitude (air density) along with other constant parameters



```
function [P_min] = minPower_cal(density, weight, wingArea, ...
    drag_polar_coeff, zeroLiftDrag_coeff)

% Simplify variables
rho = density;
W = weight;
S = wingArea;
K = drag_polar_coeff;
C_D0 = zeroLiftDrag_coeff;

% Breaking up Computation of P_min
A = 2 * W^2 ./ rho / S;
B = sqrt(3 * K^3 * C_D0);

% P_min is
P_min = 4 / 3 * sqrt(A .* B);
end
```

>> Minimum Thrust Calculator

-- This function calculates the minimum thrust as a function of the constant parameters



```
function [T_min] = minThrust_cal(altitude, weight, drag_polar_coeff, zeroLiftDrag_coeff)

% Simplify variables
W = weight;
K = drag_polar_coeff;
C_D0 = zeroLiftDrag_coeff;

% T_min is
T = 2 * W * sqrt(K * C_D0);
T_min = T .* ones(1, length(altitude));
end
```

```
function [p] = pressureCal(g, R, T, T1, h, h1, Th, unit)
```

FUNCTION DESCRIPTION: This function is designed to calculate the pressure at a where the temperature is constant in a specific altitude, such as: tropopause, stratopause, mesopause, etc.

This function is from HW #1

OUTPUT VARIABLES:

p: The pressure at an altitude h (Pa)

INPUT VARIABLES:

g: Gravitational acceleration [SI or English untis]

R: Gas Constant specific to planet [SI or English units]

T: Vector of temperatures at a certain altitude [K]

T1: Vector of average temperature at the average surface level or initial temperature [K]

h: Vector of the specific altitude [m] or [ft]

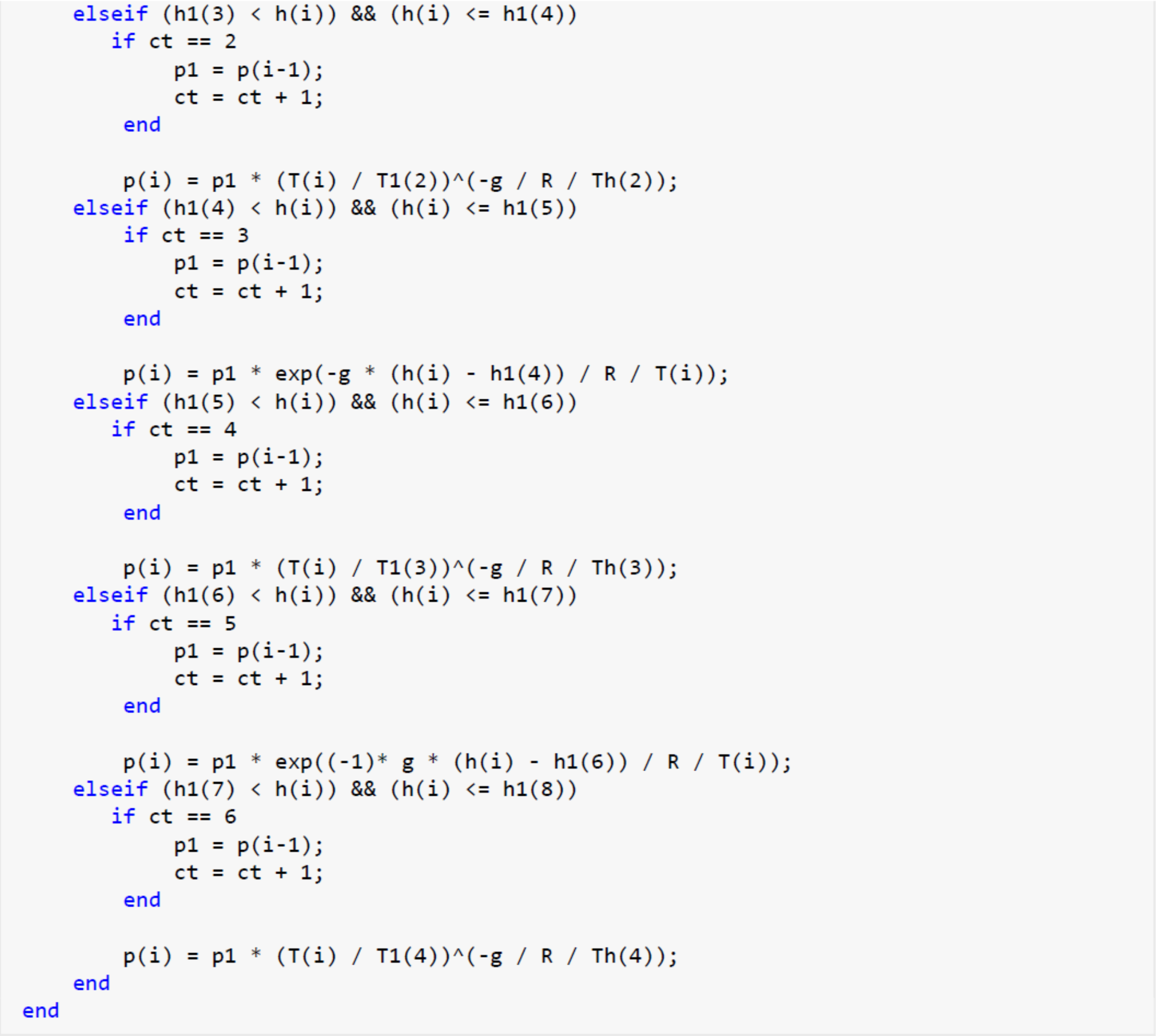
h1: Vector of the average surface level or initial surface level [m] or [ft]

Th: Vector of temperatures lapse rate

unit: String indicating English or SI units

MAIN (CODE)

```
sz = size(h);
p = zeros(sz);

if unit == "SI"
    p1 = 1013.2*100; % Initial pressure at surface (Pa)
else
    p1 = 2116.12;    % initial presure at surface (lb/ft^2)
end

ct = 0;          % counter

for i = 1:1:length(h)
    if h(i) <= h1(2)
        if ct == 0
            ct = ct + 1;
        end

        p(i) = p1 * (T(i) / T1(1))^(-g / R / Th(1));
    elseif (h1(2) < h(i)) && (h(i) <= h1(3))
        if ct == 1
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * exp(-g * (h(i) - h1(2)) / R / T(i));
```



```
    elseif (h1(3) < h(i)) && (h(i) <= h1(4))
       if ct == 2
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * (T(i) / T1(2))^(-g / R / Th(2));
    elseif (h1(4) < h(i)) && (h(i) <= h1(5))
        if ct == 3
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * exp(-g * (h(i) - h1(4)) / R / T(i));
    elseif (h1(5) < h(i)) && (h(i) <= h1(6))
       if ct == 4
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * (T(i) / T1(3))^(-g / R / Th(3));
    elseif (h1(6) < h(i)) && (h(i) <= h1(7))
       if ct == 5
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * exp((-1)* g * (h(i) - h1(6)) / R / T(i));
    elseif (h1(7) < h(i)) && (h(i) <= h1(8))
       if ct == 6
            p1 = p(i-1);
            ct = ct + 1;
        end

        p(i) = p1 * (T(i) / T1(4))^(-g / R / Th(4));
    end
end
```

```
function [T] = tempCal(T1, h, h1, Th)
```

FUNCTION DESCRIPTION: This function is designed to calculate the temperature in a specific altitude, such as: tropopause, stratopause, mesopause, etc.

This function is from HW #1

OUTPUT VARIABLES:

T: Vectors of the temperatures at an altitude h (K)

INPUT VARIABLES:

T1: Average temperature at initial level [K]

h: Vectors of the specific altitude [m or ft]

h1: Vectors of the average surface level or initial surface level [m or ft]

Th: Vectors of temperature lapse rates [K/m or K/ft]

MAIN (CODE)

```
T = zeros(size(h)); % Preallocation of temperature vector

for i = 1:1:length(h)
    if h(i) <= h1(2)
        T(i) = T1(1) + Th(1)*(h(i) - h1(1));
    elseif (h1(2) < h(i)) && (h(i) <= h1(3))
        T(i) = T1(2);
    elseif (h1(3) < h(i)) && (h(i) <= h1(4))
        T(i) = T1(2) + Th(2)*(h(i) - h1(3));
    elseif (h1(4) < h(i)) && (h(i) <= h1(5))
        T(i) = T1(3);
    elseif (h1(5) < h(i)) && (h(i) <= h1(6))
        T(i) = T1(3) + Th(3)*(h(i) - h1(5));
    elseif (h1(6) < h(i)) && (h(i) <= h1(7))
        T(i) = T1(4);
    elseif (h1(7) < h(i)) && (h(i) <= h1(8))
        T(i) = T1(4) + Th(4)*(h(i) - h1(7));
    end
end
```

>> Max Range Calculator for Propeller-driven airplane

-- this function will calculate the maximum range for an aircraft

```
function [R_prop_max] = maxRangeProp_cal(SFC, maxGrossWeight, maxFuelWeight, dragPolar_coeff..
    , zeroLiftDrag_coeff, wingArea, prop_eff)
% Assigning variables for simplicity
W_gross = maxGrossWeight; % [N] or [lb]
W_fuel = maxFuelWeight; % [N] or [lb]
K = dragPolar_coeff;
C_D0 = zeroLiftDrag_coeff;
S = wingArea;
% Converting the TSFC
c = SFC / 3600 / 550;
% Initial weight
W1 = W_gross;
% Final weight
W2 = W_gross - W_fuel;
% Lift coefficient
C_L_max = sqrt(C_D0 / K);
% drag coefficient
C_D_max = 2 * C_D0;
% Propeller efficiency
etha = prop_eff;

% Max range for propeller driven airplane
R_prop_max = (etha / c) * (C_L_max / C_D_max) * log(W1 / W2);
end
```

For this homework you may reuse as many of your Matlab functions from the previous homework as you like, but you must include the code in your response.

<u>Question 1</u>

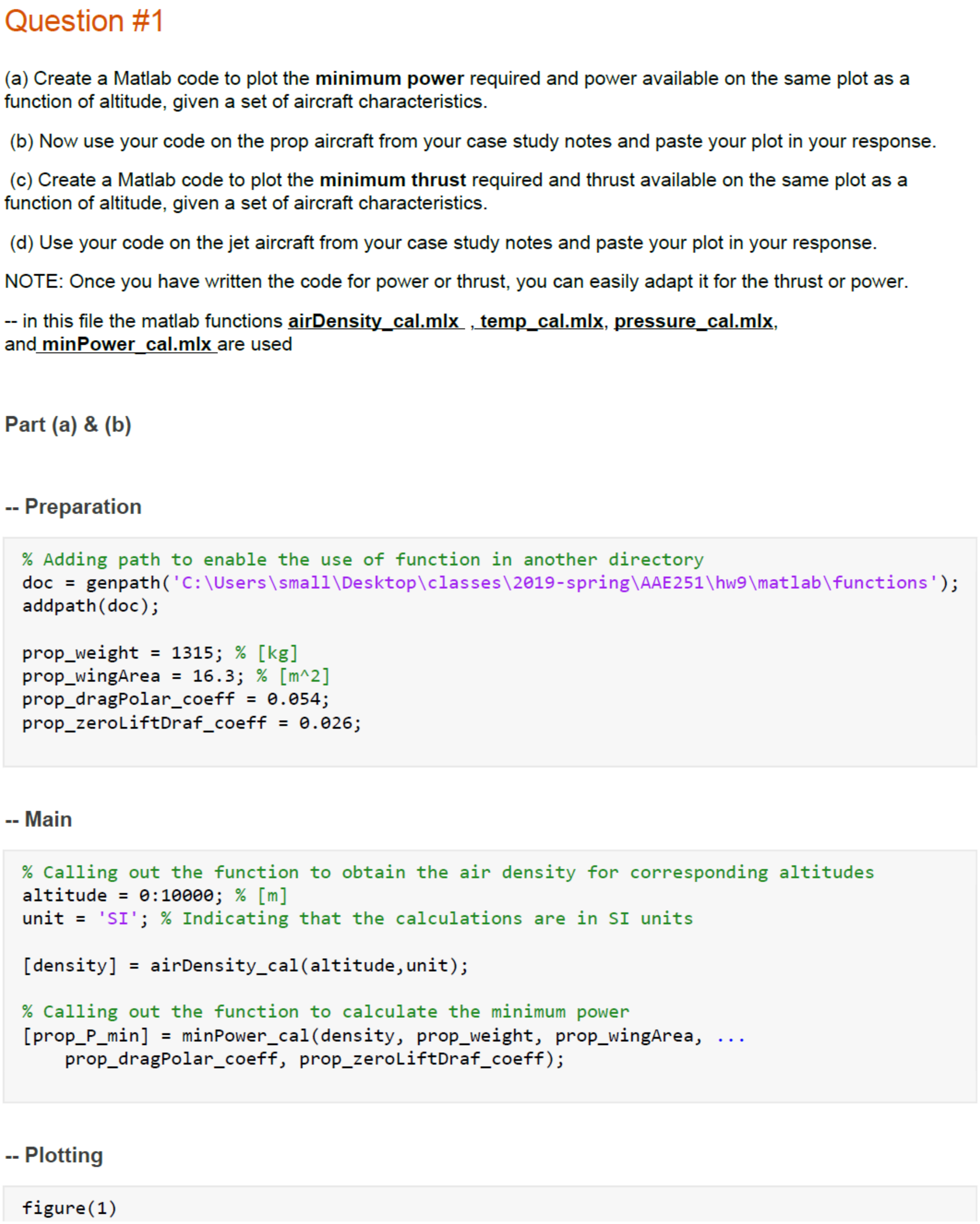
a) Create a Matlab code to plot the **minimum power** required and power available on the same plot as a function of altitude, given a set of aircraft characteristics.

b) Now use your code on the prop aircraft from your case study notes and paste your plot in your response.

c) Create a Matlab code to plot the **minimum thrust** required and thrust available on the same plot as a function of altitude, given a set of aircraft characteristics.

d) Use your code on the jet aircraft from your case study notes and paste your plot in your response.

NOTE: Once you have written the code for power or thrust, you can easily adapt it for the thrust or power.



Question #1

(a) Create a Matlab code to plot the **minimum power** required and power available on the same plot as a function of altitude, given a set of aircraft characteristics.

(b) Now use your code on the prop aircraft from your case study notes and paste your plot in your response.

(c) Create a Matlab code to plot the **minimum thrust** required and thrust available on the same plot as a function of altitude, given a set of aircraft characteristics.

(d) Use your code on the jet aircraft from your case study notes and paste your plot in your response.

NOTE: Once you have written the code for power or thrust, you can easily adapt it for the thrust or power.

-- in this file the matlab functions **airDensity_cal.mlx** , **temp_cal.mlx**, **pressure_cal.mlx**, and **minPower_cal.mlx** are used

Part (a) & (b)

-- Preparation

```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

prop_weight = 1315; % [kg]
prop_wingArea = 16.3; % [m^2]
prop_dragPolar_coeff = 0.054;
prop_zeroLiftDraf_coeff = 0.026;
```

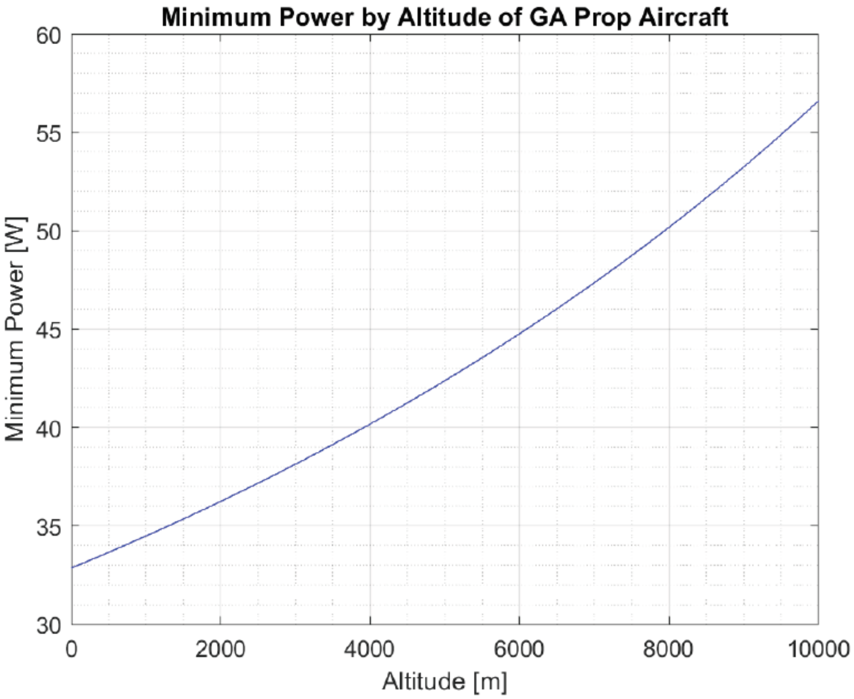
-- Main

```
% Calling out the function to obtain the air density for corresponding altitudes
altitude = 0:10000; % [m]
unit = 'SI'; % Indicating that the calculations are in SI units

[density] = airDensity_cal(altitude,unit);

% Calling out the function to calculate the minimum power
[prop_P_min] = minPower_cal(density, prop_weight, prop_wingArea, ...
    prop_dragPolar_coeff, prop_zeroLiftDraf_coeff);
```

-- Plotting

```
figure(1)
```

```
plot(altitude, prop_P_min, '-b')
title('Minimum Power by Altitude of GA Prop Aircraft')
xlabel('Altitude [m]')
ylabel('Minimum Power [W]')
grid on
grid minor
box on
```



Part (c) & (d)

-- Preparation

```
jet_weight = 33100; % [kg]
jet_wingArea = 88.2; % [m^2]
jet_dragPolar_coeff = 0.05;
jet_zeroLiftDrag_coeff = 0.015;
```

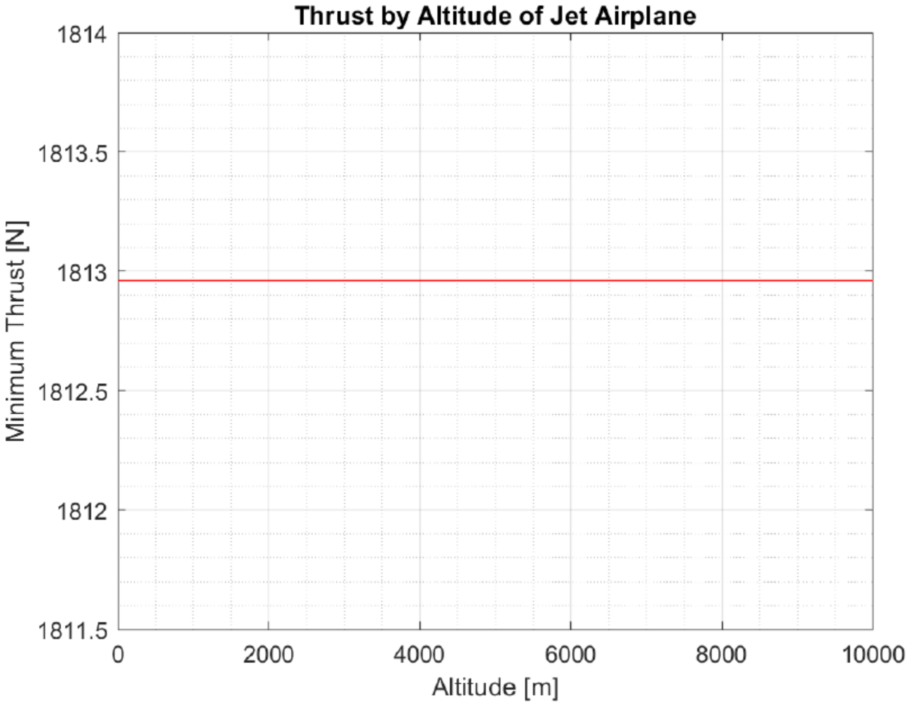
-- Main

```
% Calling out the minimum thrust calculating function
```

```
[jet_T_min] = minThrust_cal(altitude, jet_weight, jet_dragPolar_coeff,...
    jet_zeroLiftDrag_coeff);
```

Plotting

```
figure(2)
plot(altitude, jet_T_min, '-r')
title('Thrust by Altitude of Jet Airplane')
xlabel('Altitude [m]')
ylabel('Minimum Thrust [N]')
grid on
grid minor
box on
```



Question 2

The Predator UAV has a wing span of 14.85 m and a wing area of 11.5 m^2. Its maximum mass is 1020 kg. The power plant is a Rotax four-cylinder, four-stroke, engine of 85 horsepower, driving a two-blade, variable-pitch pusher propeller. Assume the Oswald efficiency factor is 0.7, the zero-lift drag coefficient is 0.03, and the propeller efficiency is 0.9. Calculate the maximum velocity of the Predator at sea level [You can use MATLAB to solve the polynomial equation].

Question #2

-- In this question the function **maxVel_for_power.mlx** will be called in order to solve the corresponding maximum velocity for the power.

Preparation

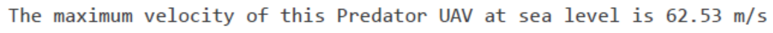
```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

wing_span = 14.85; % [m]
wing_area = 11.5; % [m^2]
mass = 1020; % [kg]
horsePower = 85; % English units
e_Oswald = 0.7;
zeroLiftDrag_coeff = 0.03;
prop_eff = 0.9;
density = 1.225; % [kg/m^3]

% converting horsepower to Watts
power = 63384.5; % [W]

% Drag polar coefficient K becomes
AR = wing_span^2 / wing_area;
dragPolar_coeff = 1 / pi / AR / e_Oswald;

% The maximum power becomes
P_max = 0.9 * power; % [W]

% the weight of the aircraft becomes
weight = mass * 9.81;
```

Main

```
% Calling out the function to get the maximum velocity
V_max = maxVel_for_power(P_max, weight, density, wing_area, ...
    zeroLiftDrag_coeff, dragPolar_coeff);
```

Result

```
fprintf(['The maximum velocity of this ',...
    'Predator UAV at sea level is %.2f m/s'], V_max);
```



```
The maximum velocity of this Predator UAV at sea level is 62.53 m/s
```

Question 3

An aircraft weighs 25,000 kg and has a wing area of 82 m^2. Its drag polar is given by

$$C_D = 0.016 + 0.04C_L^2$$

a) Find the minimum thrust required for SLUF and corresponding airspeed at sea level.
b) Find the minimum power required for SLUF and corresponding airspeed at sea level.

Question #3

-- In this question the function **minPower_cal.mlx** and **minThrust_cal.mlx** are used

Preparation

```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

wingArea = 82; % [m^2]
weight = 25000; % [kg]
zeroLiftDrag_coeff = 0.016;
density = 1.225; % [kg/m^3]
dragPolar_coeff = 0.04;
altitude = 0;
```

Main

(a)

```
% Calling out the function to calculate the minimum thrust
[T_min] = minThrust_cal(altitude, weight, dragPolar_coeff, zeroLiftDrag_coeff);
```

--result

```
fprintf(['The required minimum thrust for ', ...
    'this aircraft is %.2f N'], T_min);
```

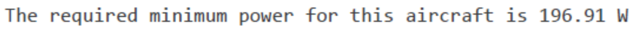
```
The required minimum thrust for this aircraft is 1264.91 N
```

(b)

```
% Calling out the function to calculate the minimum power
[P_min] = minPower_cal(density, weight, wingArea, ...
    dragPolar_coeff, zeroLiftDrag_coeff);
```

--result

```
fprintf(['The required minimum power for ', ...
    'this aircraft is %.2f W'], P_min);
```



```
The required minimum power for this aircraft is 196.91 W
```

Question 4

The Douglas DC-3 has a maximum velocity of 336 ft/s at an altitude of 7500 ft. Its engines provide a maximum power of 1200 hp. Its weight is 25,000 lb, aspect ratio is 9.14, and wing area is 987 ft^2. Assume that the propeller efficiency is 0.9, m_{ad} equal to 1, and the Oswald efficiency factor is 0.7. Calculate the zero-lift drag coefficient for the DC-3.
Density = 1.8975E-3

Q#4

$$\begin{cases} V = 336 \text{ ft/s} \\ @ \text{ alt} = 7500 \text{ ft} \rightarrow \rho = 1.8975 \times 10^{-3} \text{ slug/ft}^3 \\ hp = 1200 \\ W = 25000 \text{ lb} \\ AR = 9.14 \\ S = 987 \text{ ft}^2 \\ \eta = 0.9 \\ m_{ad} = 1 \\ e = 0.7 \end{cases}$$

$$K = \frac{1}{\pi e AR} \cong 0.04975$$

$$C_L = \frac{2W}{\rho S V^2} \cong 0.2365$$

$$q_\infty = \frac{1}{2}\rho V^2 = 107.1 \text{ lb/ft}^2$$

$$1200 \, hp \Longrightarrow P_{MAX} = 6.6 \times 10^5 \text{ lb-ft/s}$$

$$P_a = 2\eta \left(\frac{\rho}{\rho}\right)^{m_{ad}} P_{MAX} = 2 \cdot 0.9 \left(\frac{1.8975 \times 10^{-3}}{2.377 \times 10^{-3}}\right)(6.6 \times 10^5)$$

$$\cong 9.484 \times 10^5 \text{ lb-ft/s} \quad \cdots ①$$

$$C_D = C_{D0} + KC_L^2$$

$$P_a = (qSC_{D0} + qSKC_L^2)V$$

$$C_{D0} = \frac{P_a/V - qSKC_L^2}{qS}$$

$$= \frac{\left(\frac{9.484 \times 10^5 \frac{\text{lb-ft}}{s}}{336 \text{ ft/s}}\right) - (107.1 \frac{\text{lb}}{\text{ft}^2})(987 \text{ ft}^2)(0.04975)(0.2365^2)}{(107.1 \frac{\text{lb}}{\text{ft}^2})(987 \text{ ft}^2)}$$

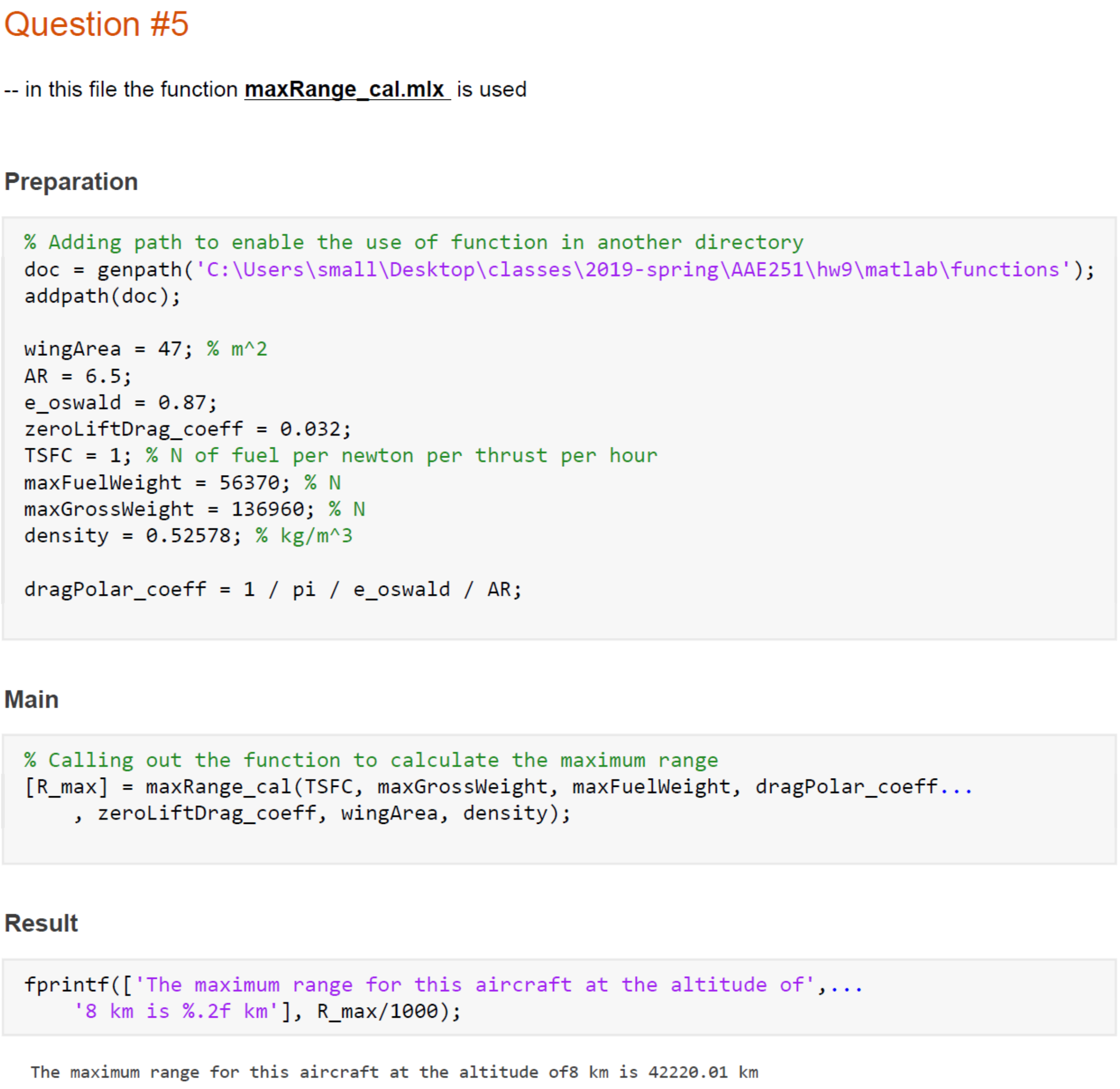
$$\cong 0.02392$$

$$\boxed{C_{D0} = 0.0239}$$

Question 5

Consider an airplane patterned after the A-10. The airplane has the following characteristics:- wing area = 47 m^2, aspect ratio = 6.5, Oswald efficiency factor = 0.87, and zero lift drag coefficient = 0.032. The TSFC is 1 N of fuel per newton of thrust per hour, the weight of fuel is 56,370 N, and the maximum gross weight is 136,960 N. Calculate the maximum range at a standard altitude of 8 km.

Density = 5.2578E-1



Question #5

-- in this file the function **maxRange_cal.mlx** is used

Preparation

```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

wingArea = 47; % m^2
AR = 6.5;
e_oswald = 0.87;
zeroLiftDrag_coeff = 0.032;
TSFC = 1; % N of fuel per newton per thrust per hour
maxFuelWeight = 56370; % N
maxGrossWeight = 136960; % N
density = 0.52578; % kg/m^3

dragPolar_coeff = 1 / pi / e_oswald / AR;
```

Main

```
% Calling out the function to calculate the maximum range
[R_max] = maxRange_cal(TSFC, maxGrossWeight, maxFuelWeight, dragPolar_coeff...
    , zeroLiftDrag_coeff, wingArea, density);
```

Result

```
fprintf(['The maximum range for this aircraft at the altitude of',...
    '8 km is %.2f km'], R_max/1000);
```

```
The maximum range for this aircraft at the altitude of8 km is 42220.01 km
```

Question 6

Consider an airplane patterned after the Beechcraft Bonanza V-tailed single-engine light private airplane. The characteristics of this airplane are as follows: aspect ratio = 6.2, wing area = 181 ft^2, Oswald efficiency factor = 0.91, and zero-lift drag coefficient = 0.027. The SFC is 0.42 lb of fuel per horsepower per hour, the weight of the fuel is 248 lb, and the maximum gross weight is 3400 lb. Calculate the maximum range at a standard sea level.

Question #6

-- in this file the function **maxRangeProp_cal.mlx** is used

Preparation

```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

wingArea = 181; % ft^2
AR = 6.2;
e_oswald = 0.91;
zeroLiftDrag_coeff = 0.027;
SFC = 0.42; % N of fuel per newton per thrust per hour
maxFuelWeight = 248; % N
maxGrossWeight = 3400; % N
prop_eff = 0.83;

dragPolar_coeff = 1 / pi / e_oswald / AR;
```

Main

```
% Calling out the function to calculate the maximum range
[R_prop_max] = maxRangeProp_cal(SFC, maxGrossWeight, maxFuelWeight, dragPolar_coeff...
    , zeroLiftDrag_coeff, wingArea, prop_eff);
```

Result

```
fprintf(['The maximum range for this aircraft at ',...
    'sea level is %.2f miles'], R_prop_max*0.000189394);
```

```
The maximum range for this aircraft at sea level is 719.04 miles
```

Question 7

An aircraft patterned after the Cessna Citation weighs 60 kN and has a wing area of 30 m^2. Its engines produce a maximum thrust of 12 kN at sea level where the air density is 1.225 kg/m^3. The drag polar is given as:

$$C_D = 0.022 + 0.047 C_L^2$$

Calculate the maximum airspeed of this aircraft when flying at sea level.

Question #7

-- In this question the function **maxVel_for_thrust.mlx** will be called in order to solve the corresponding maximum velocity for the power

Preparation

```
% Adding path to enable the use of function in another directory
doc = genpath('C:\Users\small\Desktop\classes\2019-spring\AAE251\hw9\matlab\functions');
addpath(doc);

wing_area = 30; % [m^2]
e_Oswald = 0.7;
zeroLiftDrag_coeff = 0.022;
density = 1.225; % [kg/m^3]
dragPolar_coeff = 0.047;

% The maximum thrust is
T_max = 12000; % [N]

% the weight of the aircraft becomes
weight = 60000; % [N]
```

Main

```
% Calling out the function to get the maximum velocity
V_max = maxVel_for_thrust(T_max, weight, density, wing_area, ...
    zeroLiftDrag_coeff, dragPolar_coeff);
```

Result

```
fprintf(['The maximum velocity of this ',...
    'Cessna Citation at sea level is %.2f m/s'], V_max);
```

```
The maximum velocity of this Cessna Citation at sea level is 169.99 m/s
```