Assignment 1 (10 Marks)

ENGG1001: Programming for Engineers – Semester 2, 2022

Due: Friday, 26th Aug 5pm

Introduction

An aviation authority is investigating potential sites for a new airport and they have commissioned you to do an analysis of the viability of two new sites.

The first site has an area which could accommodate a main runway which is 2 km long, but which is flat. The second site could accommodate a main runway which is 1.9 km long, but which slopes downwards from the holding area by 0.03 radians. You need to write a computer program which can determine whether either or both of these sites are suitable for the kinds of planes which normally take off from commercial airports. A model of the plane motion on the runway is given below.

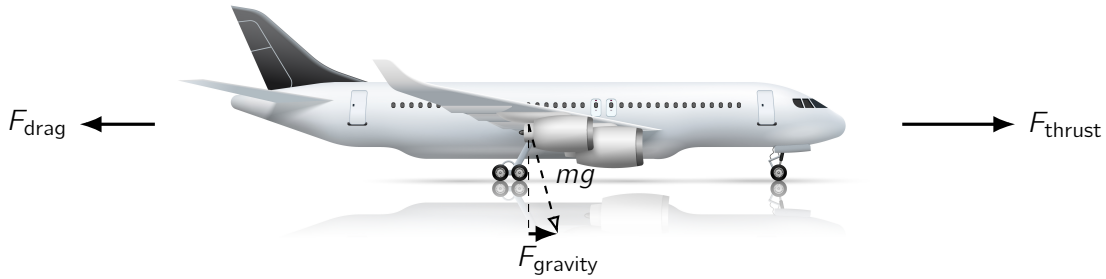

Figure 1: *Force diagram on a taxiing aircraft*

The acceleration, velocity, and position of the plane along the runway can be calculated iteratively with the following equations:

$$a_i = \frac{1}{m}\left(F_{\text{thrust}} + \overbrace{mg\sin\theta}^{F_{\text{gravity}}} - \overbrace{\frac{1}{2}\rho v_i^2 A_{\text{ref}} C_D}^{F_{\text{drag}}}\right) \quad (1)$$

$$v_{i+1} = v_i + a_i \Delta t \quad (2)$$

$$x_{i+1} = x_i + v_i \Delta t + \frac{1}{2} a_i (\Delta t)^2 \quad (3)$$

m	mass of the plane
F_{thrust}	engine thrust
g	gravity
θ	runway slope angle
A_{ref}	reference area
v_{lift}	lift off velocity
C_D	drag coefficient
ρ	air density
v_0	initial velocity at start of runway
x_0	position of the start of the runway
Δt	time increment

Where x is the distance travelled in the horizontal direction, v is the velocity in the horizontal direction, a is the acceleration in the horizontal direction, the subscript i denotes a value at t_i and

Δt is the time interval at which successive new values of x_{i+1} and v_{i+1} are computed.

It is required that you write a program which determines the trajectory of the plane along a runway with specified characteristics, and whether or not the plane is able to take off comfortably before the end of the runway (i.e. whether the plane is able to take off before reaching 85% along the runway).

Definitions
Throughout the Implementation section, three `tuples`, `runway_attributes`, `plane_attributes`, and `inputs`, will be mentioned a few times. The order of the elements within these `tuples` are as follows:

runway_attributes	(runway length, runway slope)
plane_attributes	(mass of the plane, thrust, reference area, lift off velocity, drag coefficient)
inputs	(air density, initial velocity, start position, time increment)

To assist you to structure your code the program is to be written in several stages, as outlined in the Implementation section.

1 Getting Started

Download `a1.zip` from Blackboard - this archive contains the necessary files to start this assignment. Once extracted, the `a1.zip` archive will provide the following files:

`a1.py` This is the *only* file you will submit and is where you write your code. Do not make changes to any other files.

`a1_support.py` Do not modify or submit this file, it contains functions to help you implement your tasks. You do not need to read or understand the code itself, but you should read the docstrings in order to get an idea of how the functions work. This file also contains some constants that will be useful in your implementation. In addition to these, you are encouraged to create your own constants in `a1.py` where possible.

`data/` This a folder containing example of runway and plane data files. Feel free to create your own to test the functionality of your code but keep in mind that these will not be submitted.

2 Implementation

2.1 Getting Input

```
1 def specify_inputs() -> tuple[tuple[float, ...], tuple[float, ...],
2     tuple[float, ...]]:
3     ...
```

Returns a `tuple` containing three `tuples`: the first is a `tuple` of runway attributes, the second is a `tuple` of plane attributes and the third is a `tuple` of user specified input parameters.

Example

```
>>> runway_attributes, plane_attributes, inputs = specify_inputs()
Input the length of the runway (km): 2
Input the slope of the runway (rad): 0
Input the mass of the plane (in kg): 45_000
Input the engine thrust (in N): 550_000
Input the reference area (in m^2): 750
Input the lift-off velocity (m/s): 70
Input the drag coefficient: 0.1
Input the air density (in kg/m^3): 1
Input the initial velocity (v_0) at start of runway (in m/s): 0
Input the position (x_0) at the start of the runway (in m): 0
Input the time increment (in secs): 0.1
>>> runway_attributes
(2.0, 0.0)
>>> plane_attributes
(45000.0, 550000.0, 750.0, 70.0, 0.1)
>>> inputs
(1.0, 0.0, 0.0, 0.1)
```

Note: In the Example above, the mass of the plane is $45,000kg$ and the engine thrust is $550,000N$.

2.2 Calculating Acceleration

```
def calculate_acceleration(plane_attributes: tuple[float, ...], density: float,
    velocity: float, slope: float) -> float:
    ...
```

Returns the **acceleration** calculated by the given parameters. The calculation is based on formula equation (1) in the *Introduction* section.

Example

```
>>> plane_attributes = (45_000.0, 550_000.0, 750.0, 70.0, 0.1)
>>> calculate_acceleration(plane_attributes, 1, 0, 0)
12.222222222222221
>>> calculate_acceleration(plane_attributes, 1, 0.1, 2)
21.142421646048827
>>> plane_attributes = (50000.0, 550_000.0, 800.0, 70.0, 0.1)
>>> calculate_acceleration(plane_attributes, 1, 12, 1.5)
20.670225818585777
```

2.3 Calculating Motion

```
def calculate_motion(
    runway_attributes: tuple[float, ...],
    plane_attributes: tuple[float, ...],
    inputs: tuple[float, ...]
```

```
) -> tuple[tuple[float, ...], tuple[float, ...], bool]:
    ...
```

Returns a `tuple` containing: a `tuple` of **velocities** for each successive increment of time up to the *lift velocity*, a `tuple` of **positions** for each successive increment of time up to the *lift velocity*, and **True** if the runway is *suitable* (**False** otherwise). The velocity calculations are based on formula equation equation (2) in the Introduction. The position calculations are based on formula equation equation (3) in the Introduction. The position and velocity are of the plane as it moves along the runway with each velocity and position value rounded to 3 decimal points.

Example

```
>>> velocities, positions, suitability = calculate_motion(
...     (2.0, 0),
...     (45_000.0, 550_000.0, 750.0, 70.0, 0.1),
...     (1.0, 0.0, 0.0, 0.1))
>>> velocities
(0.0, 1.222, 2.444, 3.666, 4.887, 6.107, 7.326, 8.544, 9.76, 10.975, 12.187,
 ↪  13.397, 14.604, 15.808, 17.01, 18.208, 19.403, 20.593, 21.78, 22.963, 24.141,
 ↪  25.315, 26.484, 27.647, 28.806, 29.959, 31.106, 32.248, 33.384, 34.513, 35.636,
 ↪  36.752, 37.862, 38.965, 40.06, 41.149, 42.23, 43.304, 44.37, 45.428, 46.478,
 ↪  47.52, 48.554, 49.58, 50.597, 51.606, 52.607, 53.598, 54.581, 55.555, 56.52,
 ↪  57.476, 58.423, 59.361, 60.289, 61.209, 62.119, 63.019, 63.911, 64.792, 65.665,
 ↪  66.528, 67.381, 68.225, 69.059, 69.884, 70.699)
>>> positions
(0.0, 0.061, 0.244, 0.55, 0.978, 1.527, 2.199, 2.993, 3.908, 4.945, 6.103, 7.382,
 ↪  8.782, 10.302, 11.943, 13.704, 15.585, 17.585, 19.703, 21.94, 24.296, 26.768,
 ↪  29.358, 32.065, 34.888, 37.826, 40.879, 44.047, 47.328, 50.723, 54.231, 57.85,
 ↪  61.581, 65.422, 69.373, 73.434, 77.603, 81.88, 86.263, 90.753, 95.348, 100.048,
 ↪  104.852, 109.759, 114.768, 119.878, 125.088, 130.399, 135.808, 141.314,
 ↪  146.918, 152.618, 158.413, 164.302, 170.285, 176.36, 182.526, 188.783, 195.129,
 ↪  201.564, 208.087, 214.697, 221.392, 228.173, 235.037, 241.984, 249.013)
>>> suitability
True
```

2.4 Display

```
def print_table(
    runway_test_data: tuple[tuple[float, ...], ...],
    plane_test_data: tuple[tuple[float, ...], ...],
    inputs: tuple[float, ...],
    start_column: int,
    end_column: int
) -> None:
    ...
```

This function should print out the positional data for each runway, including if the runway is suitable or not, given a `tuple` containing plane information, a `tuple` containing runway information, and a `tuple` containing input attributes. The amount of columns displayed are dictated by the `start_column` and `end_column` integers. Assume that `start_column` $\leq$ `end_column`.

Example

```
>>> plane_test_data = (
...     (45_000,550_000,750,67,0.013),
...     (50000,600000,800,70,0.05),
...     (55000,650000,900,75,0.1),
...     (60000,700000,1000,78,0.15),
...     (65000,750000,1000,80,0.234))
>>> runway_test_data = ((2.0, 0), (1.8,0.03))
>>> inputs = (1.0, 0.0, 0.0, 0.1)
>>> print_table(runway_test_data, plane_test_data, inputs, 1, 2)
#################################################################
#   Plane number    # Runway 1 distance # Runway 2 distance #
#################################################################
#         0         #     190.378     T #     188.076     T #
#         1         #     224.077     T #     222.280     T #
#         2         #     306.575     T #     298.332     T #
#         3         #     425.508     T #     403.095     T #
#         4         #    1771.301     F #     999.350     T #
#################################################################

>>> print_table(runway_test_data, plane_test_data, inputs, 1, 1)
########################################
#   Plane number    # Runway 1 distance #
########################################
#         0         #     190.378     T #
#         1         #     224.077     T #
#         2         #     306.575     T #
#         3         #     425.508     T #
#         4         #    1771.301     F #
########################################

>>> print_table(runway_test_data, plane_test_data, inputs, 2, 2)
########################################
#   Plane number    # Runway 2 distance #
########################################
#         0         #     188.076     T #
#         1         #     222.280     T #
#         2         #     298.332     T #
#         3         #     403.095     T #
#         4         #     999.350     T #
########################################

>>>
```

Hint: You should create the above table by using f-string formatting. Check the appendix for layout dimensions.

Main function

```python
def main() -> None:
    ...
```

This function handles the user interaction. It has no parameters and returns nothing. *Hint:* Make use of functions that were written in previous tasks.

Assumptions

- Commands can be incorrect.
- Runway/plane/input values or file/directory names are correct when entered.
- There will be two runways at maximum

Example 1

```
Please enter a command: h

The available commands are:
    'h' - Displays the help text.
    's' - Repleace existing runway and input parameters.
        Adds to existing plane parameters.
    's i' - to specify input parameters only. Replaces existing input
        parameters.
    'p <First runway> <Last runway>' - "p 1 2" prints out the liftoff
        distances on Runways 1 and 2; "p 1 1" prints out the liftoff distances
        on Runway 1 only.
    'r <r or p>' -
        'r r' read runway parameters from a file and replace existing runway
            parameters.
        'r p' read plane parameters from a file and add to existing plane
            parameters.
    'q' - quit.

Please enter a command: s
Input the length of the runway (km): 2
Input the slope of the runway (rad): 0
Input the mass of the plane (in kg): 45_000
Input the engine thrust (in N): 550_000
Input the reference area (in m^2): 750
Input the lift-off velocity (in m/s): 67
Input the drag coefficient: 0.013
Input the air density (in kg/m^3): 1
Input the initial velocity (v_0) at start of runway (in m/s): 0
Input the position (x_0) at the start of the runway (in m): 0
Input the time increment (in secs): 0.1
Please enter a command: p 1 1
##########################################
#   Plane number    # Runway 1 distance #
##########################################
```

```
#         0         #     190.378     T #
#########################################

Please enter a command: r p
Please specify the directory: data
Please specify the filename: plane_test_data.txt
Please enter a command: p 1 1
#########################################
#   Plane number    # Runway 1 distance #
#########################################
#         0         #     190.378     T #
#         1         #     190.378     T #
#         2         #     224.077     T #
#         3         #     306.575     T #
#         4         #     425.508     T #
#         5         #    1771.301     F #
#########################################

Please enter a command: r r
Please specify the directory: data
Please specify the filename: runway_test_data.txt
Please enter a command: p 1 1
#########################################
#   Plane number    # Runway 1 distance #
#########################################
#         0         #     190.378     T #
#         1         #     190.378     T #
#         2         #     224.077     T #
#         3         #     306.575     T #
#         4         #     425.508     T #
#         5         #    1771.301     F #
#########################################

Please enter a command: p 1 2
#################################################################
#   Plane number    # Runway 1 distance # Runway 2 distance #
#################################################################
#         0         #     190.378     T #     188.076     T #
#         1         #     190.378     T #     188.076     T #
#         2         #     224.077     T #     222.280     T #
#         3         #     306.575     T #     298.332     T #
#         4         #     425.508     T #     403.095     T #
#         5         #    1771.301     F #     999.350     T #
#################################################################

Please enter a command: k
Please enter a valid command.
Please enter a command: q
Are you sure (y/n): y
```

Example 2

```
Please enter a command: r r
Please specify the directory: data
Please specify the filename: runway_test_data.txt
Please enter a command: r p
Please specify the directory: data
Please specify the filename: plane_test_data.txt
Please enter a command: p 1 1
Please enter the parameters first.
Please enter a command: s i
Input the air density (in kg/m^3): 1
Input the initial velocity (v_0) at start of runway (in m/s): 0
Input the position (x_0) at the start of the runway (in m): 0
Input the time increment (in secs): 0.1
Please enter a command: p 1 2
#################################################################
#   Plane number    # Runway 1 distance # Runway 2 distance #
#################################################################
#         0         #     190.378     T #     188.076     T #
#         1         #     224.077     T #     222.280     T #
#         2         #     306.575     T #     298.332     T #
#         3         #     425.508     T #     403.095     T #
#         4         #    1771.301     F #     999.350     T #
#################################################################

Please enter a command: r p
Please specify the directory: data
Please specify the filename: plane_test_data.txt
Please enter a command: p 1 2
#################################################################
#   Plane number    # Runway 1 distance # Runway 2 distance #
#################################################################
#         0         #     190.378     T #     188.076     T #
#         1         #     224.077     T #     222.280     T #
#         2         #     306.575     T #     298.332     T #
#         3         #     425.508     T #     403.095     T #
#         4         #    1771.301     F #     999.350     T #
#         5         #     190.378     T #     188.076     T #
#         6         #     224.077     T #     222.280     T #
#         7         #     306.575     T #     298.332     T #
#         8         #     425.508     T #     403.095     T #
#         9         #    1771.301     F #     999.350     T #
#################################################################

Please enter a command: q
Are you sure (y/n): n
Please enter a command:
```

Writing your code

You must download the file, **a1.py**, from Blackboard, and write your code in that file. When you submit your assignment to Gradescope **you must submit only one file and it must be called**

a1.py. Do not submit any other files or it can disrupt the automatic code testing program which will grade your assignments.

Design

You are expected to use good programming practice in your code, and you must incorporate at least the functions described in the previous part of the assignment sheet.

Assessment and Marking Criteria

The maximum achievable mark for this assignment is 10 marks.

Functionality Assessment — 7 Marks

The functionality will be marked out of 7. Your assignment will be put through a series of tests and your functionality mark will be based on how your program performs on the automatic tests. If, say, there are 25 functionality tests and you pass 20 of them, then your functionality mark will be $20/25 \times 7$. You will be given the functionality tests before the due date for the assignment so that you can gain a good idea of the correctness of your assignment yourself before submitting. You should, however, make sure that your program meets all the specifications given in the assignment. That will ensure that your code passes all the tests. Note: Functionality tests are automated and so string outputs need to exactly match what is expected. Do not leave it until the last minute because it generally takes time to get your code to pass the tests.

Code Style Assessment — 3 Marks

The style of your assignment will be assessed by one of the tutors, and you will be marked according to the style rubric provided with the assignment. The style mark will be out of 3.

Assignment Submission

You must submit your completed assignment to Gradescope. The only file you submit should be a single Python file called **a1.py** (use this name — all lower case). This should be uploaded to Gradescope. You may submit your assignment multiple times before the deadline — only the last submission will be marked.

Late submission of the assignment will **not** be accepted. In the event of exceptional personal or medical circumstances that prevent you from handing in the assignment on time, you may submit a request for an extension. See the course profile for details on how to apply for an extension.

Requests for extensions **must** be made according to the guidelines in the ECP.

Appendix A: Layout Dimensions

Table dimensions:

```
#################################################################
#   Plane number    # Runway 1 distance # Runway 2 distance #
#################################################################
#         0         #     190.378     T #     188.076     T #
```

```
#         1         #     224.077     T #     222.280     T #
#         2         #     306.575     T #     298.332     T #
#         3         #     425.508     T #     403.095     T #
#         4         #    1771.301     F #     999.350     T #
#####################################################################
  |<-- 19 chars -->|  |<-- 19 chars -->|  |<-- 19 chars -->|
```