FIT9133 Assignment #1
Simulating a Town

Semester 1 2019

Gavin Kroeger
Teaching Associate, Faculty of IT
Email: Gavin.Kroeger@monash.edu

Assignment Structure by Jojo Wong

August 7, 2019

Revision Status

```
$Id: A1.tex, Version 1.1 2019/07/04 3:30 pm Gavin Kroeger $
$Id: A1.tex, Version 1.2 2019/07/08 2:30 pm Gavin Kroeger $
$Id: A1.tex, Version 1.3 2019/07/17 2:30 pm Gavin Kroeger $
$Id: A1.tex, Version 1.4 2019/08/03 9:50 pm Gavin Kroeger $
$Id: A1.tex, Version 1.5 2019/08/05 9:56 pm Gavin Kroeger $
$Id: A1.tex, Version 1.6 2019/08/07 6:23 am Gavin Kroeger $
$Id: A1.tex, Version 1.6 2019/08/07 6:58 pm Gavin Kroeger $
```

Contents

1 Introduction **4**

2 Simulating a Town **5**

2.1 Overview 5

2.2 The Assignment 6

2.2.1 Task 1: Reading and writing data 6

2.2.2 Task 2: Calculating Food 7

2.2.3 Task 3: Calculating Population 8

2.2.4 Task 4: Performing the Simulation 9

3 Important Notes **10**

3.1 Documentation 10

3.2 Marking Criteria 10

4 Submission **11**

4.1 Deliverables 11

4.2 Academic Integrity: Plagiarism and Collusion 11

1 Introduction

This assignment is due on **8th September 11:55 pm**. It is worth **15% of the total unit marks**. A penalty of 10% per day will apply for late submission. Refer to the FIT9133 Unit Guide for the policy on extensions or special considerations.

Note that this is **an individual assignment** and **must be your own work**. Please pay attention to Section 4.2 of this document on the university policies for the *Academic Integrity, Plagiarism and Collusion*.

All the program files and any supporting documents should be compressed into one single file for submission. (The submission details are given in Section 4.)

Note: for the sake of this assignment you do not need to import any libraries. However, if you want to, you may import **math**.

2 Simulating a Town

2.1 Overview

Your assignment is to simulate life in an average Australian country town. Your simulation must incorporate the following characteristics:

- On average the population increases by 30% each year due to immigration and natural births.
- Every 18 years the population is reduced by 40% due to children coming of age and being drafted for football. Year 0 is considered a draft year. The calculation for draft numbers is done after the calculation for population growth.
- Each unit of food is enough to feed a single person for an entire year.
- People are allocated their food at the start of each year. If for some reason there isn't enough food for a person they will leave town and go to the city.
- Due to many factors, 1/5 of all food must be thrown away at the end of every year before the new food arrives.
- Food to feed 80% of the population in the form of meat pies, sausage rolls, and potato cakes arrives at the end of each year after the old food has been thrown away. The food order is placed as the last action for the town for the year. This ensures that the most amount of people are catered for.

Normally your simulation should run for 100 years, however if at the start of a year the town's population is 0, the simulation ends early. The starting characteristics of the town should be read from a file called *town_start.txt*. The ending state of the town should be saved to a file called *town_end.txt*.

Be sure to read the specification very carefully. Your code will be partially marked based on its output which must match those given in this specification.

NOTE: Please ensure that all calculations are rounded *down* (or floored) as it is against most laws to store half a person, and no one in this town leaves food partially eaten.

2.2 The Assignment

2.2.1 Task 1: Reading and writing data

Write a script called *{YourStudentID}_task1.py* that reads in the town starting data. This data should be stored appropriately named variables.

The starting file will be called *town_start.txt* and will have the following format:

- Starting food
- Starting population
- Starting year

Your python file should then create a new file called *town_end.txt*, writing the data you just read. It will have the following format:

- Remaining food
- Remaining population
- Final year

Examples of these files are provided on Moodle. Note that for this task the numbers will not change from reading to writing but will change in future tasks. Use the following data to test your code:

	Input	Output
Example 1	100 100 0	100 100 0
Example 2	10 100 0	10 100 0

Table 1: Test data for Task 1. Note it follows the format of Food, Population, and Year.

2.2.2 Task 2: Calculating Food

Write a script called *{YourStudentID}_task2.py* that calculates the food remaining at the end of a single year's cycle. You do not need to consider changes in population in this script. Use the techniques from Task 1 to again read and write the appropriate files.

	Input	Output
Example 1	100 100 0	80 100 1
Example 2	10 100 0	80 100 1
Example 3	100 10 0	80 10 1
Example 4	1000 10 0	800 10 1
Example 5	155 90 0	124 90 1

Table 2: Test data for Task 2. Note it follows the format of Food, Population, and Year.

2.2.3 Task 3: Calculating Population

Write a script called *{YourStudentID}_task3.py* that calculates the population at the end of a single year's cycle. Do not include changes to food here, only the changes to the population. You must consider if the year is a multiple of 18 for the sake of drafting new football players.

	Input	Output
Example 1	100 100 0	100 78 1
Example 2	10 100 0	10 7 1
Example 3	100 10 0	100 7 1
Example 4	1000 10 0	1000 7 1
Example 5	155 90 0	155 70 1

Table 3: Test data for Task 3. Note it follows the format of Food, Population, and Year.

2.2.4 Task 4: Performing the Simulation

Using the techniques demonstrated in the previous tasks, create the full simulation of the town in a script called *{YourStudentID}_task4.py*. Your simulation should adhere to the following algorithm which was created using the overview in section 2.1:

1. Food is handed out to the town.
2. People leave if there is no food for them.
3. Food is sorted and rotten food is thrown away.
4. Population grows.
5. Every 18 years, a draft takes place.
6. The food order is placed and food is delivered for next year.

	Input	Output
Example 1	100 100 0	108 136 100
Example 2	10 100 0	0 0 22
Example 3	100 10 0	0 0 76
Example 4	1000 10 0	156 195 100
Example 5	155 90 0	213 267 100

Table 4: Test data for Task 4. Note it follows the format of Food, Population, and Year.

3 Important Notes

3.1 Documentation

Commenting your code is essential as part of the assessment criteria (refer to Section 3.2). You should also include comments at the beginning of your program file, which specify your name, your Student ID, the start date, and the last modified date of the program, as well as with a high-level description of the program. In-line comments within the program are also part of the required documentation.

3.2 Marking Criteria

The assessment of this assignment will be based on the following marking criteria:

- 60% for working program — functionality;
- 10% for code architecture — algorithms, data types, control structures, and use of libraries;
- 10% for coding style — clear logic, clarity in variable names, and readability;
- 20% for documentation — program comments.

4 Submission

There will be NO hard copy submission required for this assignment. You are required to submit your assignment as a **.zip** file name with your Student ID. For example, if your Student ID is **12345678**, you would submit a zipped file named "**A1_12345678.zip**". Note that marks will be deducted if this requirement is not strictly complied with.

Your submission must be done via the assignment submission link on the FIT9133 S2 2019 Moodle site by the deadline specified in Section 1. Submissions will not be accepted if left in Draft mode. Be sure to submit your files before the deadline to not incur late penalties.

4.1 Deliverables

Your submission should contain the following documents:

- Four Python scripts named "**{YourStudentID}_task1.py**", "**{YourStudentID}_task2.py**", "**{YourStudentID}_task3.py**", and "**{YourStudentID}_task4.py**".

  NOTE: Your programs must at least run on the computers in the University's computer labs. Any submission that does not run accordingly will receive no marks.

- Electronic copies of ALL your files that are needed to run your programs.

Your programs must be written as python scripts. Jupyter Notebooks will not be accepted as part of submission for this assignment.
Marks will deducted for any of these requirements that are not strictly complied with.

4.2 Academic Integrity: Plagiarism and Collusion

Plagiarism Plagiarism means to take and use another person's ideas and or manner of expressing them and to pass them off as your own by failing to give appropriate acknowledgement. This includes materials sourced from the Internet, staff, other students, and from published and unpublished works.

Collusion Collusion means unauthorised collaboration on assessable work (written, oral, or practical) with other people. This occurs when you present group work as your own or as the work of another person. Collusion may be with another Monash student or with people

or students external to the University. This applies to work assessed by Monash or another university.

It is your responsibility to make yourself familiar with the University's policies and procedures in the event of suspected breaches of academic integrity. (Note: Students will be asked to attend an interview should such a situation is detected.)

The University's policies are available at: `http://www.monash.edu/students/academic/policies/academic-integrity`