FIT9133 Assignment #1

Semester 1 2019

Gavin Kroeger
Admin Tutor, Faculty of IT
Email: Gavin.Kroeger@monash.edu

Assignment Structure by Chunyang Chen

March 19, 2019

Revision Status

```
$Id: FIT9133-Assignment-01.tex, Version 1.0 2019/02/20 19:30 pm Gavin Kroeger $
$Id: FIT9133-Assignment-01.tex, Version 1.2 2019/02/25 21:30 pm Gavin Kroeger $
$Id: FIT9133-Assignment-01.tex, Version 1.3 2019/03/01 10:57 pm Chunyang Chen $
$Id: FIT9133-Assignment-01.tex, Version 1.3 2019/03/18 01:25 pm Chunyang Chen $
```

Contents

1 Introduction

This assignment is due on **Apr 12, 2019 (Friday) 5pm**. It is worth **15%** of the total unit marks. **No late submissions** will be accepted. 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*. You will be interviewed by the tutors in the lab after the assignment due, to test if you really understands the solution to the assignment.

This first assignment consists of **Three** main tasks and each task should be submitted as a separate Python program with supporting documentation on its usage. Please write your code in the template files provided along with the PDF description. 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.)

2 Calculating Skill

A nerd is a person seen as overly intellectual, obsessive, introverted or lacking social skills. To check if a person is nerd or not, you can examine many aspects of this person. For your assignment you must create a functioning program that allows individuals to calculate their 'nerd skill level'. The higher the skill level, the more powerful the nerd. This level is calculated using a special formula found in section 2.2. You will be asked to implement a menu to allow the program to be used, an equation which can be used to calculate the score, and a printout that allows a summation of the results.

You will need to make use of the Math library when implementing Task 2.

Other than Math, you may make use of the following Python built-in libraries if you wish:

- String
- Random

Note: You MAY use these libraries. However, their use is not needed to complete the assignment! You should **NOT make use of** any other third-party libraries to complete this assignment.

2.1 Task 1: A Menu

This menu is the main point of interaction between the user and the program. After each option is chosen, your program should return to the menu, so the user can select another option. All the input from users should be stored in variables and please check the validity of the input.

Your menu must have the following options:

- Enter Fandom Score
  - A person's 'Fandom Score' is the number of things that the person considers themself a 'fan of'. For example, if a person likes both Star Wars and Star Trek but nothing else, then they have a score of 2.
  - Note that a Fandom Score must be non-zero and non-negative. If a person enters a bad number the program should output an error message, but continue to run.
- Enter Hobbies Score
  - A person's 'Hobbies Score' is the number of hobbies a person undertakes on a **weekly** basis. For example, if a person plays Fortnite, sews, dances, and swims every week, then they have a Hobbies Score of 16 (4 x 4) as we assume that one month contains 4 weeks in this assignment.
  - Note that a Hobbies Score must be multiples of 4. Once the input is not valid, an error message must be displayed.
  - Further, 0 is the multiple of 4 for the sake of this assignment.
- Enter Number of Sports played
  - A person is considered to play a sport if they own an item that could be used in that sport. If a person owns a soccer ball then their score is 1. If an item can be used for multiple sports, then it only counts as 1 towards this number.
  - This score must be positive. Once again, error messages should appear if it is entered as negative.
- Calculate Nerd Score
  - This menu option will calculate the person's Nerd Score. The score should be printed to the screen when calculated. The score is calculated using Figure 1.
  - If any of the numbers above are not entered, then this option should display an error telling the user what it is missing.
- Print Nerd Rating of Students

– Display the results of Nerd Class of a list of students. The class is determined by Table 2.

NOTE: You should write your code in the template file "**menu_StudentID.py**" provided in the Moodle, and please change the StudentID into your own ID.

2.2 Task 2: Implement the Nerd Score

$$x\sqrt{\frac{42y^2}{z+1}}$$

Figure 1: Where x = Fandom Score, y = Hobbies Score, and z = Number of Sports Played

After obtaining the input value from the last task, you must implement the equation as shown in Figure 1. You must store the required variables appropriately and place the equation in your menu. The user should be able to get their Nerd Score, change one of the variables, and then recalculate to get the new value. For the sake of testing, use Table 1 below.

x	y	z	Nerd score
1	4	1	18.33030277982336
20	4	1	366.6060555964672
20	20	1	1833.030277982336
1	20	1	91.6515138991168
1	4	50	3.6299408518921203

Table 1: Example values of x,y,z and their calculated results.

NOTE: You should write your code in the template file "**nerdScore_StudentID.py**" provided in the Moodle, and please change the StudentID into your own ID.

2.3 Task 3: Finding your Class

The numbers that come from the equation are mostly useless except for bragging rights. As such your program should output the person's Nerd Class which is determined by the score. Once a score surpasses a classes threshold, then they are considered to be a part of that class. For example, if the author of this paper received a score of 98.0, then he would be classed as a "Nerdlinger". However if the author received a score of 100.0, then he would be classed as a "Nerd".

Score	Class	Position in the output list
0	Nerdlite	0
1	Nerdling	1
10	Nerdlinger	2
100	Nerd	3
500	Nerdington	4
1000	Nerdrometa	5
2000	Nerd Supreme	6

Table 2: Score thresholds and their associated Class.

The target of this task is that given several students' score (stored in a list), we hope to output the detailed numbers of each nerd class (stored in a list) defined in Table 2. Each index of the output list represent different class as mentioned in Table 2, e.g., the number of students who are categorized as "Nerdlite" lies in the first item of the output list, i.e., the index of "Nerdlite" in the output list as 0. One test case is that given a list of nerd score [23, 76, 1300, 600], your program should output a list [0, 0, 2, 0, 1, 1, 0]. This list signifies that there are:

- 0 Nerdlite
- 0 Nerdling
- 2 Nerdlinger
- 0 Nerd
- 1 Nerdington
- 1 Nerdrometa
- 0 Nerd Supreme

NOTE: you should write your code in the template file "**findClass_StudentID.py**" provided in the Moodle, and please change the StudentID into your own ID.

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. The same marking criteria will be applied on both tasks:

- 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 and user documentation.

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 S1 2019 Moodle site by the deadline specified in Section 1, and **No delay is allowed** unless very spacial reasons with medical certificate.

4.1 Deliverables

Your submission should contain the following documents:

- At least the Three Python scripts named as "**menu_StudentID.py**", "**skillEquation_StudentID.py**" and **findClass_StudentID.py**". If you have more than Three scripts for your implementation, name them with the same convention.
- A report (in the form of a PDF) that demonstrates how to use your program through the use of text and screenshots. You must detail which choices you made to extend your program.

  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.

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`