**Self-Reflection/Self Marking Sheet**

**Repository Link**: https://drive.google.com/open?id=0B4Ug9aaD5rq5dFYySlpaWlRQeFU

1) Support command-line arguments

* **Ownership:** My own work.
* **Self- reflection and Robustness:** The program allows command line arguments and uses “welcome” as one so that a welcome message and the date is displayed each time the cmd is opened. This feature is robust enough to handle any script parameter and if it is not valid a “\*\*\* Unknown syntax” message appears then the cmd continues to run.
* **Expected mark:** 1/1.5

2) Has a line-oriented command interpreter based on cmd

* **Ownership:** My own work.
* **Self- reflection and Robustness:** The program created uses cmd based interpreter to handle user input and output. I believe that I have implemented feature successfully as it is robust enough to handle a user input entered into it without breaking. I could have implemented this differently with the use of an interface if I had more time available.
* **Expected mark:** 1/1.5

3) Display command line help of available commands

* **Ownership:** My own work.
* **Self- reflection and Robustness:** All cmd commands have useful help commands available which explain how each command works and how to use it. The help command only shows the commands that have help provided and each of them have a description of what the command does and how to use it.
* **Expected mark:** 1/1.5

4) Change options

* **Ownership:** My own work.
* **Self- reflection and Robustness:** The cmd commands allow various options to them such as the load being able to choose between “load file” and “load database” or the display function being able to choose between, “display unchecked”, display stored” and “display graph”. These extra options have validations in place to ensure robustness of the program so I believe this feature has been completed to a high standard.
* **Expected mark:** 1/1.5

5) Validate your selections

* **Ownership:** My own work.
* **Self- reflection and Robustness:** When users input their selections such as described in the previous point, their input is validated to ensure on valid data is being passed through. Various checks have been put in place in case the user adds any extra invalid data when stating a selection.
* **Expected mark:** 1/1.5

6) Provides object-persistence/ object serialization using either pickle or shelve

* **Ownership:** My own work.
* **Self- reflection and Robustness:** Pickle has been used to serialize the data that has been stored after it has been checked. This feature is robust due to the fact that it can only be executed when the input data is loaded, validated and saved. It allows the user to be able to send the stored data to a pickle file and also to read from it.
* **Expected mark:** 1/1.5

7) Load data from a file

* **Ownership:** My own work.
* **Self- reflection and Robustness:** The user is able to load sets of data via .txt files. In the CMD, load is followed by “file [directory]” to retrieve the data and store it. Validation has been put in place for program robustness so that only valid files and directories can be entered and executed.
* **Expected mark:** 1/1.5

8) Raises exceptions and provides exception handling

* **Ownership:** My own work.
* **Self- reflection and Robustness:** The use of exceptions is used throughout my code to catch any errors that might occur that could break the code. These are mainly used in my controller as that is where most of the users input is handled. Many different exceptions such as IndexError, AttributeError and ValueError have been used to catch any issues ensure the code does not break during usage.
* **Expected mark:** 1/1.5

9) Amount of error trapping & handling

* **Ownership:** My own work.
* **Self- reflection and Robustness:** Throughout my code I have used various error handling techniques to avoid breaking the code so it can run smoothly and display and errors that occurred in a user friendly way. The many uses of this feature have helped to ensure I have a robust program. Uses such as checking if the variables value has changed since initialisation or the use of Booleans for checks and flags have allowed me to build a high level of robustness throughout.
* **Expected mark:** 1/1.5

10) Provide doctests

* **Ownership:** My own work.
* **Self- reflection and Robustness:** Doctests have been created and used to test if the load function in my controller works as intended. Five different examples of different doctests have been provided prove the robustness of my program. I implemented enough doctests to cover the different types of inputs my load function could handle.
* **Expected mark:** 1/1.5

11) Provide unit tests

* **Ownership:** My own work.
* **Self- reflection and Robustness:** Unit-tests have been implemented to test the outputs received from valid and invalid inputs. Thirteen different examples of unit tests have been provided and are working successfully to prove the robustness of my program.
* **Expected mark:** 1/1.5

12) Breadth of test coverage

* **Ownership:** My own work.
* **Self- reflection and Robustness:** A wide range of tests have been implemented on the program. These include:
  + Valid and Invalid inputs for the load, save, validate, welcome and display functions.
  + Two test data files, one covering many possible invalid inputs and another with fully valid input.
  + These tests add up to 18 different tests and cover most aspects of the program.
* **Expected mark:** 1/1.5

13) Can deal with directories and file locations

* **Ownership:** My own work.
* **Self- reflection and Robustness:** When loading a file, the directory or the name of the file (if it is in the same directory) can be called and its contents will successfully be stored. If an invalid file name is used, checks have been put in place to catch any exceptions such as a FileNotFoundError exception.
* **Expected mark:** 1/1.5

14) Pretty print, i.e., displaying data in bar chart, pie chart, etc.

* **Ownership:** My own work.
* **Self- reflection and Robustness:** I have implemented the plotly package to create a pie graph showing the number of male’s vs number of female’s that have been stored. This feature still has robustness so that if no data is entered, a graph will not be created to be view.
* **Expected mark:** 1/1.5

15) Can save and read data from a database

* **Ownership:** My own work.
* **Self- reflection and Robustness:** A database view is created using sqlite3 and data can be retrieved and stored using the “load database” entry in the CMD. This feature has been implemented in such a way to avoid issues such as multiple openings of the database or accidental closure of the database.
* **Expected mark:** 1/1.5

**Total Expected Mark: 15/15**