# **Owner ship and self-reflection**

## **Owner ship**

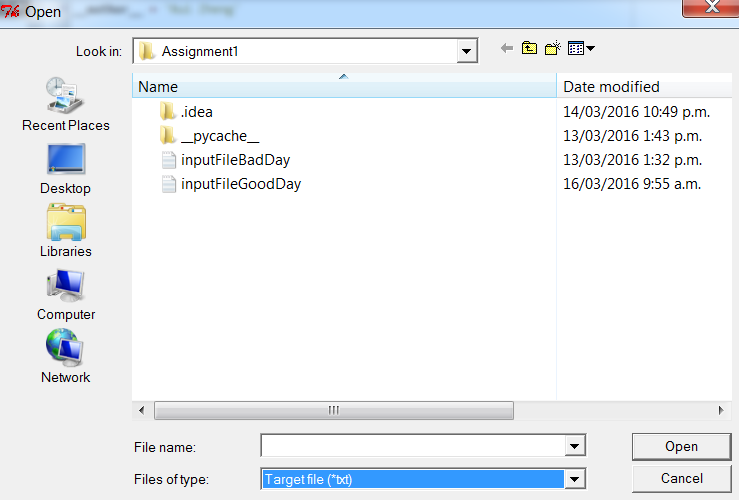
All the Components of this project are developed by Rui Zheng (10226052)

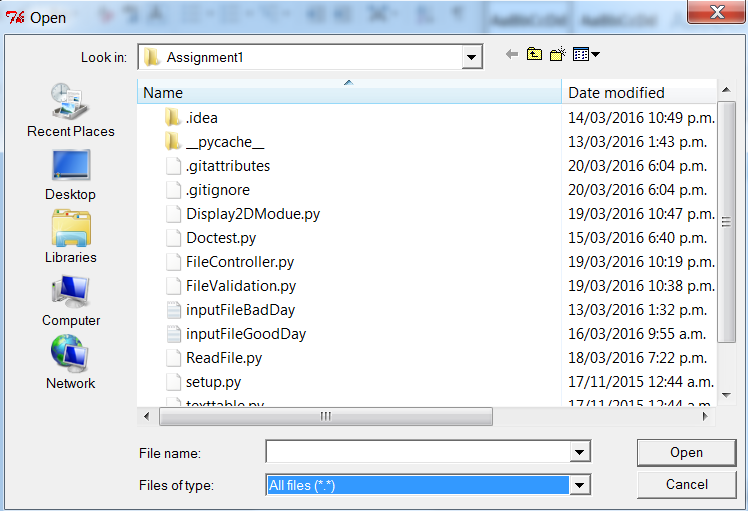
## **Self-reflection**

1. **Load data from a File**

**Robustness**

Target file will have ‘txt’ suffix, any other file will not be displayed unless toggle to the option of all file type.





**Completeness and Implementation**

It allows user to use the GUI rather than command line input, I think it will reduce the typo and other man made problems. I have a class holding the data of the file it read from previous file open function. In this case it will not ask user to load the file for validation or display 2D charts.



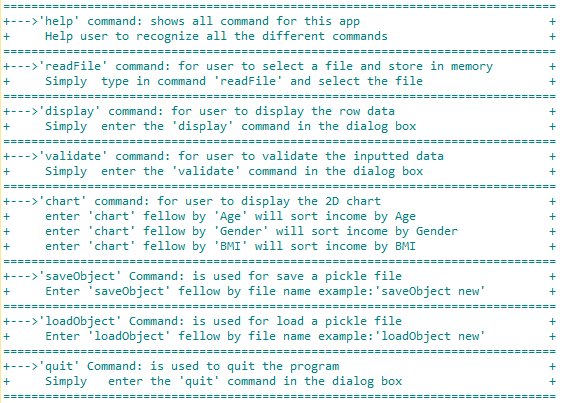
1. **display command line help of available commands.**

**Robustness**

Works every time, and have not thinking a way that may lead it to fail.

**Completeness and Implementation**

I have modified the Help command which as the build in function, it will display all the help information all at once instead of using ‘help readFile’ to display it one by one.



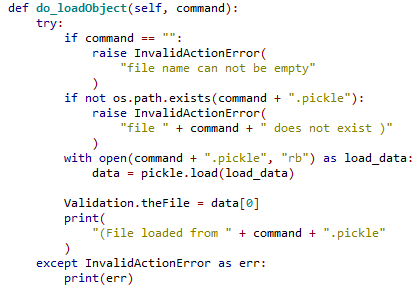
1. **Has a line-oriented command interpreter based on cmd.**

**Robustness**

CMD implemented in controller class. All class performance successful under the CMD implementation It will not allow user to interaction with the system if there is an error in the command, and also it will out put a costumed error message to let the user to the right direction.

**Completeness and Implementation**

It been by using try and catch the as error handling. The fellow example shows how it going to work.



1. **Change options**

**Robustness**

The program should take the user to the main prompt, allow user to type in for the next command with no problem. I have been modifying the and testing the functions. It seems to working just fine.

**Completeness and Implementation**

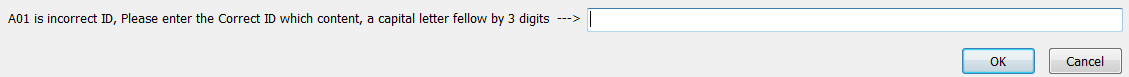
1. **Validate your selections**

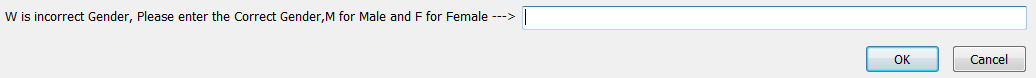
**Robustness**

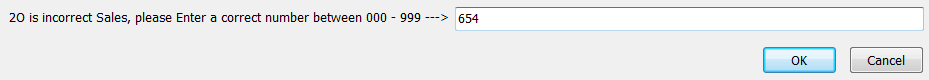
Validate the file given, it starts looking from first attribute ‘ID’ and fellow by ‘Gender’, ‘Age’, ‘Sales’, ‘BMI’ and ‘Income’. All the attributes have to fit with the regular expression that tutor has been asked for.

**Completeness and Implementation**

I have mad this works as rise the prompt to ask user to give the correct attribute toward to the incomplete or wrong attributes. There is a problem with it which it will not validate the user input, in some case the user input may still not be compatible with the given rules.







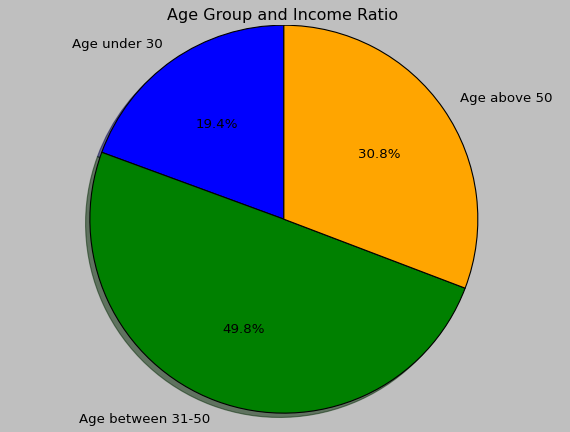
1. **Pretty-print**

**Robustness**

The data it took from validated file. Sometime user like to run the command validate before even load the data file. This program can handle it as well, as the validate class will call the read file class and ask for the input file and then validate it. The validated data then sent to the display chat class to make it happen

**Completeness and Implementation**

I think the way I did is good as it not limited users from any command. The pretty print will take 3 different command such as ‘Age’, ‘Gender’ and ‘BMI’. This will show the users what is the relation between those this with ‘Income’.



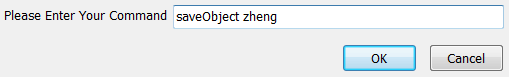
1. **Provides object-persistence / object serialization using either pickle or shelve.**

**Robustness**

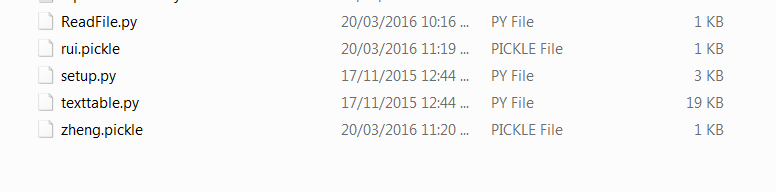
Stander object-persistence by using pickle, it asking for a file name and it will persistence of the object under the file\_name.pickle.

**Completeness and Implementation**

Using exactly same way which in class demo, only down side would be it will rewrite the file if given a same file name.







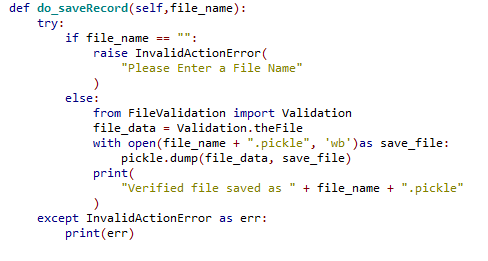
1. **Raises exceptions and provides exception handling.**

**Robustness**

The exception will catch most of error and out put the message that we have been set in the function. User can learn the program by play with it.

**Completeness and Implementation**

I have use the try and catch to handle the errors, like to following shows



1. **Amount of error trapping & handling.**

**Robustness**

should be good for the most situation, amount of ’if’, and ‘elif’ to help before any error occur. The most functions will have try and except to deals when error occurs.

**Completeness and Implementation**

Standard way to apply the functions. The amount of error trapping and handling are decent.



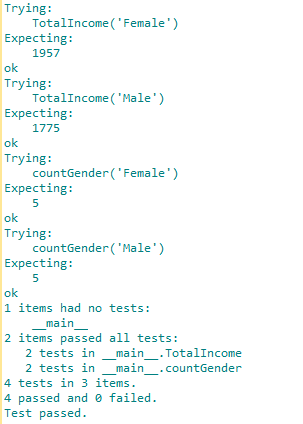
1. **Provide doctests.**

**Robustness**

Dactests base on the file been read, the file has to be the particular one that I have been calculated beforehand. It will be testing readFile as well as data collecting.

**Completeness and Implementation**

The way I did it is going to test multiple function all at one go, sometimes it may confuse user wen it not working out. As more than one function has been involved.



1. **Can deal with directories and file locations**

**Robustness**

It is very good as it using the GUI and allow user to read file form any directory.

**Completeness and Implementation**

GUI been used instated of CLI

