Brian Christopherson

18-JUN-2025

Python 100

Assignment08

https://github.com/cb658/IntroToProg-Python-Mod08

An employee review rating application using multiple code modules – Using and supporting the Employee Ratings application

Introduction

This console application allows a user to track employee review ratings. The application stores the employee's first and last names, their review date and their numeric rating (1 through 5) in the JavaScript Object Notation (JSON) format.

The core functions of the application are displaying the existing employee ratings, adding new employee ratings, and saving the new data to the existing ratings JSON file.

Using the application

To launch the application, a user has two options:

- Use the PyCharm application and navigate to the folder containing the code files. Open the main.py file and click the Run button in the PyCharm toolbar, or right click and select *Run 'main'*, or press Shift + F10
- Open a command prompt and navigate to the folder containing the code files. Type ***python main.py*** and press Enter

When the application is launched the user is presented with the Employee Ratings menu shown in Figure 1 below.

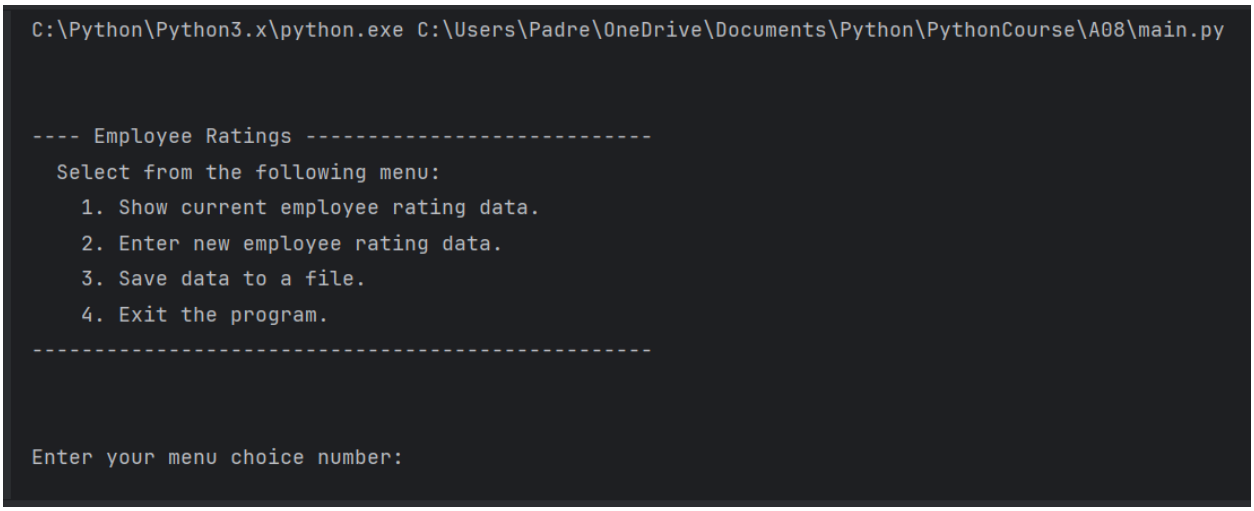

Figure 1 - The application menu

To control the application the user enters the numeral associated with their desired choice (1, 2, 3, or 4). The application choices are shown in Figure 1 above and explained in more detail in the sections below.

Menu Choice 1 – Showing current employee rating data:

To show current employee ratings, enter menu choice "1". This will show the data that is read from the EmployeeRatings.json file. If it is the first time running the application and/or no employee rating data has been previously saved (i.e. the file is empty), an error will be generated. Thus, it is recommended not to use this option until data has been saved.

If employee rating data is present in the file, menu choice 1 will display the employee's first and last names and their rating number, along with the text descriptor for that rating. The information is presented on a single line per employee. The text descriptors for the ratings are shown in the table below:

Table 1

Value	***Rating***
1	Not Meeting Expectations
2	Building
3	Solid
4	Strong
5	Leading

Figure 2 below shows the presentation of the employee ratings using example data.

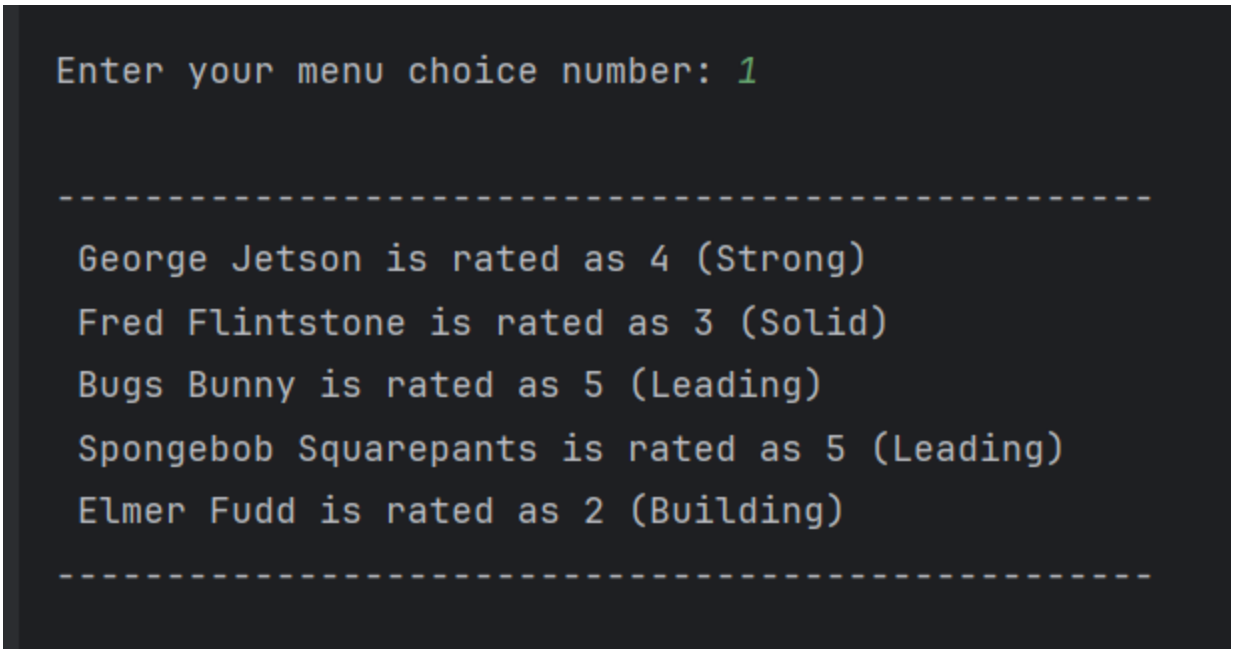

Figure 2 - Menu Choice 1, using example data

Menu Choice 2 – Entering new employee rating data:

To enter a new employee rating, enter menu choice "2". This will prompt for the employee's first name. Type the employee's first name then press Enter. Next, enter the employee's last name and press Enter. Note that it a numeric value is accidentally typed in for either name, the application will return an error that the value is not the correct type of data. An additional technical error message appears to explain that the name should not contain numbers.

Once the employee's first name and last name have been entered, the application will next ask for the employee's review date. The review date will need to be entered in the ISO format, YYYY-MM-DD. If the review date is not entered in this format, the application will return an error that the value is not the correct type of data. An additional technical error message will appear to explain that the data format is incorrect and should be YYYY-MM-DD.

When the review date is entered in the correct date format, the application will then ask for the employee's review rating. Enter a number between 1 and 5. Consult Table 1 above for a brief description of the rating numbers. Valid rating numbers are 1, 2, 3, 4 or 5. If any other value is entered, the application will advise to enter a valid number.

When all the employee rating data has been added, the existing employee rating data will be displayed again, with the newest employee rating appearing at the bottom of the list. A prompt has been added to notify the user the data has not yet been saved

Figure 3 below shows the entry prompts for menu choice 2 and the resulting output.

```
Enter your menu choice number: 2
What is the employee's first name? Tweety
What is the employee's last name? Bird
What is their review date? 1999-01-01
What is their review rating? 3

--------------------------------------------------
 George Jetson is rated as 4 (Strong)
 Fred Flintstone is rated as 3 (Solid)
 Bugs Bunny is rated as 5 (Leading)
 Spongebob Squarepants is rated as 5 (Leading)
 Elmer Fudd is rated as 2 (Building)
 Tweety Bird is rated as 3 (Solid)
--------------------------------------------------

The data has not yet been saved, please select menu choice 3 to save the new data!


---- Employee Ratings ----------------------------
```

Figure 3 - Menu choice "2"

Menu Choice 3 – Saving data to a file

When menu choice “3” is chosen, the updated employee data is saved to the EmployeeRatings.json file in JSON format. If there are any issues with the data format or permissions to the file, errors will be raised by the application. Otherwise, the data will be saved and a message will be presented indicating the data was saved to the file. Figure 4 below shows the information presented when menu choice 3 is selected.

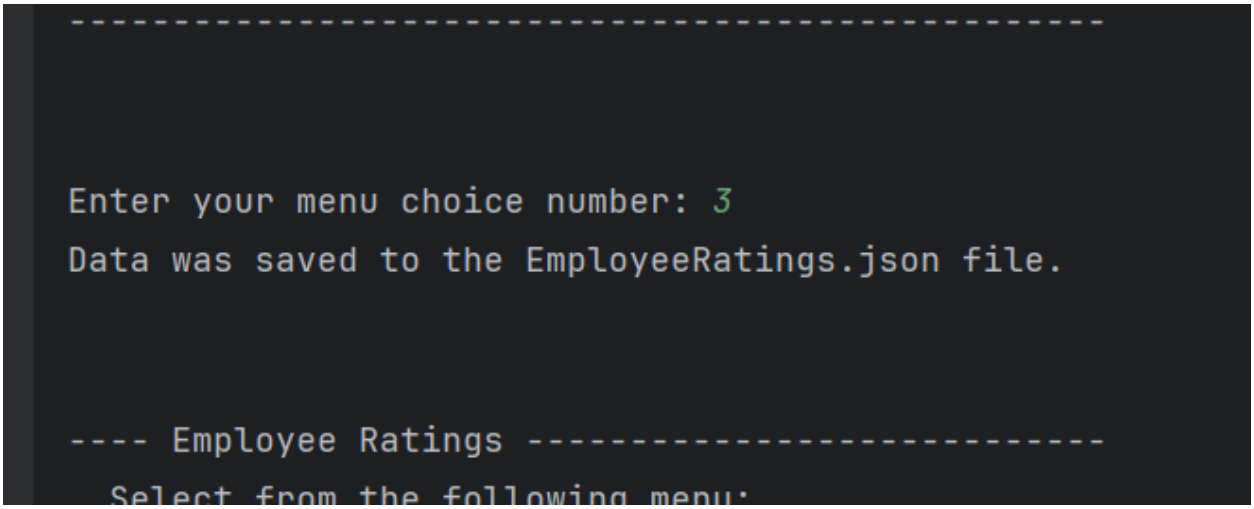

Figure 4 - Menu Choice 3

Menu Choice 4 – Exiting the program

Selecting menu choice "4" exits the program. The message "Exiting" will be displayed as the program exits. Figure 5 shows the result when menu choice 4 is chosen.

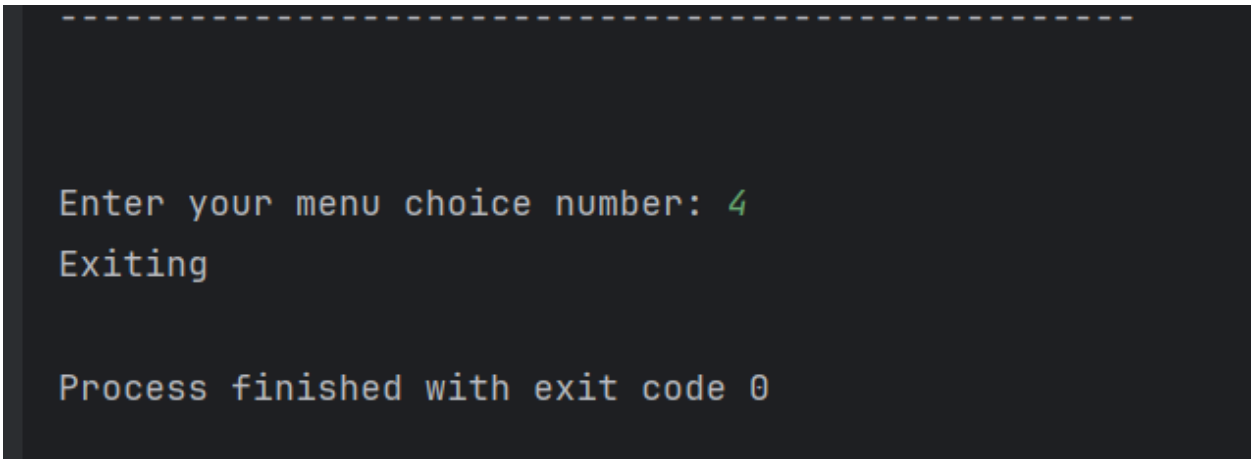

Figure 5 - Menu Choice 4

Technical Data and Application Support

There are seven files associated with the application. The files are described briefly here, and in more detail in the sections below. Aligning with the Separation of Concerns principles, the application's variables and constants are stored in the data_classes.py file. The data processing layer is contained within the processing_classes.py file, which addresses the reading from file and writing data to file. The user interface is defined within

the presentation_classes.py file. Units tests of the application's classes and functions are contained within three test files: test_data_classes.py, test_processing_classes.py, and test_presentation_classes.py; as their names imply, these test the data_classes.py, processing_classes.py, and presentation_classes.py files, respectively. Additional structured error-handling is contained within each of the primary modules.

main.py

The primary file of the application is main.py, and it launches the application. It begins with imports of the json and datetime modules, along with imports of the other modules. Aliases are associated with each module to reduce the amount of code and reduce complexity.

A Try-Except loop begins the program and sets the values of the employees list variable to the employee_data list values read in from the EmployeeRatings.json file. This is done using the ***read_employee_data_from_file*** function contained within the FileProcessor class contained in processing_classes.py. The "except" clauses reference the custom error message format contained within the ***IO.output_error_messages*** function within presentation,py.

A *while True* loop begins and calls the ***output_menu*** function from presentation_classes.py to display the MENU string constant to the user. The menu_choice variable is then set to the user input via the ***input_menu_choice*** function from presentation_classes.py.

If the menu_choice variable is set to "1", the ***output_employee_data*** function from presentation_classes.py is called to display the data contained in the employee_data list.

If the menu_choice variable is set to "2", the ***input_employee_data*** function from presentation_classes.py is called. This function accepts the employee type object inputs from the user (FirstName, LastName, ReviewDate and ReviewRating) and appends the input to the employee_data list.

If the menu_choice variable is set to "3", the ***write_employee_data_to_file*** function from processing_classes.py is called. This function writes the data to the EmployeeRatings.json file (as defined within the FILE_NAME constant). A print statement indicates to the user that the data was saved.

If the menu_choice variable is set to "4", a *break* statement is called to exit the application.

data_classes.py

The data_classes.py defines the application constants, **FILE_NAME** and **MENU**, which is the string presented to the user as the application menu. Additionally, the application

variables **employees** (a list) and **menu_choice** (a string) are defined and initially set as empty. The classes **Person** and **Employee** are defined and properties set and initialized. The __str__() method is used to return the properties in comma-separated format.

processing_classes.py

The processing_classes.py file contains a single class named **FileProcessor**, with two functions defined, ***read_employee_data_from_file*** and ***write_employee_data_to_file.*** The ***read_employee_data_from_file*** function uses json.load to populate a list of dictionary rows using each employee object in the list. These objects are then appended to the employee_data list and returned.

The ***write_employee_data_to_file*** converts the list of employee objects to a list of dictionary rows and writes the list of dictionary data to the EmployeeRatings.json file using the *json.dump* function imported from the json module.

presentation_classes.py

The presentation_classes.py file contains a single class named **IO**, which contains the user interface logic functions as described previously within this document:

- output_error_messages
- output_menu
- input_menu_choice
- output_employee_data
- input_employee_data.

Unit testing modules

Unit testing is provided by the _test modules listed below, which test the related modules.

- test_data_classes.py: this module tests the **Person** and **Employee** classes from data_classes.py
- test_processing_classes.py: this module tests the two read/write functions within the **FileProcessor** class from processing_classes.py
- test_presentation_classes.py: this module tests the two input functions within the **IO** class contained within the presentation_classes.py file

Data storage

As indicated previously, the data used by this application is contained within the EmployeeRatings.json file. The data uses the JSON format, for potential interchanges with any future web-based systems.