NOTES

Programming Assignment Using Python

By:

Pasumarthy Krishna Bharat

Completed & Submitted On:

31st October 2021

Table of Contents:

1. Introduction

This assignment is about storing and maintaining Notes created by user. Named this stored notes in a file called MyNotesLibrary.

User will be able to perform various actions in this process as below:

- Create Notes
- View Notes
- Update Notes
- Remove Notes
- Display Details of Notes
- Save Notes to MyNotesLibrary.txt file
- Load Notes to memory from MyNotesLibrary.txt

A menu control is created for the user to interact with the program and perform required action.

2. Description

In order to make it user friendly, we have created ***Classes*** which can enable us to code using Object Oriented Programming.

Classes Created:

- Main
- Note
- UserInterface
- Utilities

Let us discuss in details what each of the able class consists of, how they are used and how they are interconnected.

Class Main:

- Main Class consists of the program which allows user to view the menu required, to interact with the UserInterface in order to perform required actions.
- This is the part of program which has to be executed, which will import required modules and perform actions requested by user.

Class Note:

- Note Class gives the structure how a note has to be created, stored and what all details are required for an object to be a Note.
- Apart from required private attributes, we have
    - A counter (cntr) variable for unique id generation when a new note is created
    - A note_id_list where all the id's are stored in a list which can be used while performing various operations.
- For the attributes, accessor and mutator methods are created so that values assigned to the attributes can be accessed and updated from outside the class as per user's requirement.
- Additional methods are also created to fetch the complete information of the note from outside the class.

Class UserInterface:

- UserInterface (UI) Class enables the program to take control from the input option selected by the user from the menu and invokes corresponding function which has to be executed in order to perform the action.
- Separate methods are created for each action to be performed like creation, updation and deletion of a note and so on.

- There is a separate class called Utilities, which has methods created to execute part of similar code to be performed in various methods of UserInterface.

Class Utility:

- Utility Class has various methods which help to validate id, date and update menu when user wished to update information in a note.
- It also has the key methods which will perform the actual update actions based on user input on the note objects performing various validations to make the information in a note look meaningful.
- For example, if a note is created and status was incomplete when it was created, and now using the update option if user wants to change status to completed, then program automatically provides control to update completion date also at that instant which can avoid user to navigate to completion date again from menu.

Additional Validations and Features incorporated:

- Note creation date and completion date cannot be greater than today's date and completion date cannot be before creation date.
- User interface is created in such a way that user inputs are validated and only desired inputs will be accepted and invalid inputs are handled to guide the user what can be the input value.
- User can decide if he/she wants to update all attributes of the note or specific attribute alone through a menu option provided in UI.
- While displaying the overall details of notes created a prompt is created to decide whether user wishes to view all the notes or not.
- In some required places sleep function is used to provide a small time gap for the user to view next options.

3. User Manual

- Execute the main function (Main.main())to start with program.
- Menu options will be displayed from 1-9 and 0, to let user decide what action is to be performed.
- User Input 1: Program prompts to allow user to give title, text, status of completion of notes if it is complete or incomplete. If status is complete, then it automatically considers current date as completion date.
- User Input 2: Firstly displays list of notes available in the memory and prompts to let user enter note id he wants to view.
- User Input 3: Firstly displays list of notes available in the memory and prompts to let user enter note id he wants to update. Another prompt to decide if all attributes are to be updated or only required attribute is to be updated. Based on the user input prompt will be generated and user has to enter updated values respectively.
- User Input 4: Firstly displays list of notes available in the memory and prompts to let user enter note id he wants to delete.
- User Input 5: Displays number of notes created and shows how much percentage of notes are yet to be completed. Also prompts a message if user would like to view all the existing notes at a time.
- User Input 6: Firstly displays list of notes available in the memory and prompts to let user enter note id he wants to give completion date.
- User Input 7: Firstly displays list of notes available in the memory and prompts to let user enter note id he wants to view the time taken to complete the notes in days.
- User Input 8: Saves all the notes in memory to MyNotesLibrary.txt file in a readable format.

- User Input 9: Loads all the notes present in MyNotesLibrary.txt file to the memory and makes it accessible for the user to perform required tasks.
- User Input 10: Prompts if user would like to exit with or without saving the notes to file.

  All the user input option 2-8 will check if any notes are present in the memory and then proceeds further with required action if notes are present only.