Hyperiondev

TASK

Capstone Project - Lists, Functions, and String Handling

Visit our website

Introduction

WELCOME TO THE LISTS, FUNCTIONS, AND STRING HANDLING CAPSTONE PROJECT!

Congratulations on making it this far! This Capstone is a milestone in your learning. This project will consolidate the knowledge that you've learned across various tasks. In this project, you will be extending the functionality of the previous Capstone Project. Be creative — you'll be tasked with a set of criteria to meet but the rest is up to you! Remember, it is worth putting some extra time and effort into this project — it can become part of your developer portfolio.

DEVELOPER PORTFOLIO

Developers who have the edge are those who find ways to apply their newfound skills from the get-go. A **developer portfolio** (a collection of online creations that you have made) allows you to demonstrate your skills rather than just telling people about them. It's a way of bringing your CV to life and introducing yourself to the world. As you learn more skills and put these into practice, each project that you complete will become more efficient and eye-catching.

This application series offers you the means to create projects for your very own developer portfolio, allowing you to walk away from this course not only with a certificate but, more importantly, with a headstart into your career!

THE TASK AT HAND

In the previous Capstone project, you created a simple task management system that stored data in text files. In this project, you will use lists or dictionaries and functions to extend the functionality of your previous Capstone Project.

Before you begin

A key focus of this project will be ensuring that your code is correct, well-formatted, and readable. In this regard, make sure that you do the following before submitting your work:

1. Make sure that you have identified and removed all syntax, runtime, and logical errors from your code.

2. Make sure that your code is modular. Create functions to perform specific units of work.

3. Make sure that your code is readable. To ensure this, add comments to your code, use descriptive variable names, and make good use of whitespace and indentation.

4. Make sure that your code is as efficient as possible. How you choose to write code to create the solution to the specified problem is up to you. However, make sure that you write your code as efficiently as possible.

5. Make sure that all output that your program provides to the user is easy to read and understand. Labelling all data that you output (whether in text files or to the screen) is essential to make the data your program produces more user-friendly. For example, compare the readability of the outputs in the images below. Notice how using spacing and labelling the output makes the second output much more user-friendly than the first:

Output 1:

```
admin, Register Users with taskManager.py, Use taskManager.py to add the usernam
es and passwords for all team members that will be using this program., 10 Oct 2
019, 20 Oct 2019, No
admin, Assign initial tasks, Use taskManager.py to assign each team member with
appropriate tasks, 10 Oct 2019, 25 Oct 2019, No
```

Versus Output 2:

```
----------------------------------------------------------

Task:                   Assign initial tasks
Assigned to:            admin
Date assigned:          10 Oct 2019
Due date:               25 Oct 2019
Task Complete?          No
Task description:
 Use taskManager.py to assign each team member with appropriate tasks
----------------------------------------------------------
```

Compulsory Task 1

Follow these steps:

- Create a copy of the files (**task_manager.py**, **user.txt** and **tasks.txt)** from your previous Capstone project. In this task, you will be modifying this program.

- Modify the code of your previous project so that functions are used. Adding functions will improve the modularity of your code. Your program should include at least the following functions:

    - `reg_user` — that is called when the user selects '`r`' to register a user.

    - `add_task` — that is called when a user selects '`a`' to add a new task.

    - `view_all` — that is called when users type '`va`' to view all the tasks listed in '**tasks.txt**'.

    - `view_mine` — that is called when users type '`vm`' to view all the tasks that have been assigned to them.

- Modify the function called `reg_user` to make sure that you don't duplicate usernames when you add a new user to **user.txt**. If a user tries to add a username that already exists in **user.txt**, provide a relevant error message and allow them to try to add a user with a different username.

- Add the following functionality when the user selects '`vm`' to view all the tasks assigned to them:

    - Display all tasks in a manner that is easy to read. Make sure that each task is displayed with a corresponding number that can be used to identify the task.

    - Allow the user to select either a specific task (by entering a number) or input '`-1`' to return to the main menu.

    - If the user selects a specific task, they should be able to choose to either mark the task as complete or edit the task.

        - If the user chooses to mark a task as complete, the '`Yes`'/'`No`' value that describes whether the task has been completed or not should be changed to '`Yes`'.

        - If the user chooses to edit a task, the username of the person to whom the task is assigned or the due date of the task can

be edited. The task can only be edited if it has not yet been completed.

- Add an option to generate reports to the main menu of the application. The menu for the admin user should now look something like this:

```
Please select one of the following options:
r - register user
a - add task
va - view all tasks
vm - view my tasks
gr - generate reports
ds - display statistics
e - exit
```

- When the user chooses to generate reports, two text files, called **task_overview.txt** and **user_overview.txt**, should be generated. Both these text files should output data in a user-friendly, easy to read manner.

    - **task_overview.txt** should contain:
        - The total number of tasks that have been generated and tracked using the **task_manager.py.**
        - The total number of completed tasks.
        - The total number of uncompleted tasks.
        - The total number of tasks that haven't been completed and that are overdue.
        - The percentage of tasks that are incomplete.
        - The percentage of tasks that are overdue.

    - **user_overview.txt** should contain:
        - The total number of users registered with **task_manager.py**.
        - The total number of tasks that have been generated and tracked using **task_manager.py**.
        - For each user also describe:
            - The total number of tasks assigned to that user.
            - The percentage of the total number of tasks that have been assigned to that user
            - The percentage of the tasks assigned to that user that have been completed
            - The percentage of the tasks assigned to that user that must still be completed
            - The percentage of the tasks assigned to that user that has not yet been completed and are overdue

- Modify the menu option that allows the admin to display statistics so that the reports generated are read from **task_overview.txt** and

user_overview.txt and displayed on the screen in a user-friendly manner. If these text files don't exist (because the user hasn't selected to generate them yet), first call the code to generate the text files.

Rate us

Share your thoughts

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we've done a good job?

Click here to share your thoughts anonymously.