HyperionDev

OOP – Synthesis

Task

Visit our website

Introduction

In this task, you will apply your previously acquired knowledge of object-oriented programming (OOP) to devise a practical solution for a real-world problem. It's important to note that this task will not introduce any new concepts. Instead, it offers a valuable opportunity to review, practice, and hone your existing skills, making it a worthy addition to your developer portfolio.

The task at hand

Let us assume that you work as a store manager for a Nike warehouse. As a store manager, you are responsible for managing the warehouse, and more importantly, performing stock-taking. To optimise your delivery time and for improved organisation, you have decided to use your Python knowledge to get an overview of what each stock-taking session entails.

Nike warehouses store the following information for each stock-taking list:

- Country
- Code
- Product
- Cost
- Quantity
- Value

Other store managers (in other regions) would like to be able to use your program to do the following:

- Search products by code.
- Determine the product with the lowest quantity and restock it.
- Determine the product with the highest quantity.
- Calculate the total value of each stock item. The total value is calculated by multiplying the cost by the quantity for each item entered into the system.

Take note

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 readable. To ensure this, add comments to your code, use descriptive variable names and make good use of whitespace and indentation. Use the **PEP 8 style guide** to see how classes and methods should be named and how your program should be formatted.
3. Make sure that your code is modular. Create functions to perform specific units of work.
4. 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. Use defensive coding to validate user input and make provisions for errors that may occur using exception-handling techniques.
6. 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.

   Output 1: Raw

```
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
```

   Output 2: User-friendly

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

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

Practical task

For this task, use the file template provided named **inventory.py**.

1. Code a program that will read from the text file **inventory.txt** and perform the following on the data to prepare for presentation to your managers:
2. Inside this file, you will find a class named `Shoe` with the following attributes:
    - `country`
    - `code`
    - `product`
    - `cost`
    - `quantity`
3. Inside this class define the following methods:
    - `get_cost` – Returns the cost of the shoes.
    - `get_quantity` – Returns the quantity of the shoes.
    - `__str__` – This method returns a string representation of a class.
        - Read more about the __str__ **special method** if you are unsure how to implement it
4. Outside this class create a `shoes_list` variable with an empty list. This variable will be used to store a list of shoe objects.
5. Then you must define the following functions outside the class:
    - `read_shoes_data` – This function will open the file **inventory.txt** and read the data from this file, then create a shoe object with this data and append this object to the shoe list. One line in this file represents data to create one object of shoes. You must use the `try-except` blocks in this function for error handling. Remember to skip the first line using your code.
    - `capture_shoes` – This function will allow a user to capture data about a shoe and use this data to create a shoe object and append this object inside the shoe list.
    - `view_all` – This function will iterate over the shoe list and print the details of the shoes returned from the `__str__` function. (**Optional**: you can organise your data in a table format by using Python's `tabulate` module.)
    - `re_stock` – This function will find the shoe object with the lowest quantity, which are the shoes that need to be restocked. Ask the user if they want to add this quantity of shoes and then update it. This quantity should be updated on the file for this shoe.
    - `search_shoe` – This function will search for a shoe from the list using the shoe code and return this object so that it will be printed.

- `value_per_item` – This function will calculate the total value for each item. Please keep the formula for value in mind; `value = cost * quantity`. Print this information on the console for all the shoes.
- `highest_qty` – Write code to determine the product with the highest quantity and print this shoe as being for sale.

6. Now create a menu that executes each function above. This menu should be inside the while loop. Be creative!

Important: Be sure to upload all files required for the task submission inside your task folder and then click "Request review" on your dashboard.

Share your thoughts

Please take some time to complete this short feedback **form** to help us ensure we provide you with the best possible learning experience.
