Homework 7
Object Oriented Programming
Deadline: 07/29/2024

Instructions:
Submit your code along with explanations and test results that demonstrate the implementation of the required features.
Submit a single .pdf document in Canvas with both code and results.
Always remember, "Practice makes perfect!"

Exercise 1: (50 points) Meal and Nutrition Tracker
Objective: Develop code that allows users to log daily food intake, track nutritional values, and monitor their diet against personal health goals, supporting informed food choices and dietary objectives.

Classes and Components:
1. FoodItem
Variables: name (private), calories (private), proteins (private), carbs (private), fats (private)
Instance Methods:
__init__(self, name, calories, proteins, carbs, fats): Constructor to initialize a new food item with nutritional info.
Getter Methods for each private variable and a display method to print the food item.

2. DailyLog
Variables: date (private), food_items (a list of FoodItem instances, private)
Instance Methods:
__init__(self, date): Constructor to initialize a new daily log.
add_food_item: Adds a FoodItem instance to the log.
get_total_calories: Calculates total calories consumed on that day.
get_total_nutrients: Calculates total proteins, carbs, and fats consumed.
display: Print the daily log.

3. NutritionProfile
Variables: user_id (private), daily_logs (a dictionary with dates as keys and DailyLog instances as values)
Instance Methods:
__init__(self, user_id): Constructor to initialize a new nutrition profile.
add_daily_log(self, daily_log): Adds a DailyLog instance to the profile.
get_log_by_date(self, date): Retrieves a DailyLog by date.
display: Print the nutrition profile.

4. Testing:
Example:
```
# Create some food items
apple = FoodItem("Apple", 95, 0.5, 25, 0.3)
banana = FoodItem("Banana", 105, 1.3, 27, 0.3)

# Create a daily log and add food items
daily_log = DailyLog("2023-04-02")
daily_log.add_food_item(apple)
daily_log.add_food_item(banana)
```

```
# Create a nutrition profile and add the daily log
profile = NutritionProfile("User1")
profile.add_daily_log(daily_log)

# Testing outputs using display
apple.display()
banana.display()
daily_log.display()
profile.display()
```

Expected Output:

```
FoodItem: Apple, Calories: 95, Proteins: 0.5g, Carbs: 25g, Fats: 0.3g
FoodItem: Banana, Calories: 105, Proteins: 1.3g, Carbs: 27g, Fats: 0.3g

Daily Log: 2023-04-02
FoodItem: Apple, Calories: 95, Proteins: 0.5g, Carbs: 25g, Fats: 0.3g
FoodItem: Banana, Calories: 105, Proteins: 1.3g, Carbs: 27g, Fats: 0.3g

Nutrition Profile: User1
Log Date: 2023-04-02
Daily Log: 2023-04-02
FoodItem: Apple, Calories: 95, Proteins: 0.5g, Carbs: 25g, Fats: 0.3g
FoodItem: Banana, Calories: 105, Proteins: 1.3g, Carbs: 27g
```

Exercise 2: (50 points) Create a module

Objective: Save the classes FoodItem, DailyLog, and NutritionProfile into a Python file named nutrition_tracker.py. This file will act as your module and import it to another jupyter notebook to produce the same output as in Exercise 1.

Exercise 3: (10 points) BONUS – Plot the daily log.

Explore the package matplotlib and create a bar plot from the daily log. Implement the plot as another method, example: daily_log.plot_nutrients()

Example: