(420-P24-AB) Programming III

Winter 2021

Quiz – Practical Portion

(Out of 24 marks, worth 60% of Quiz mark or 12% of final grade)

Due Friday, Apr 23 by midnight via Lea. 0% if late (so submit early!)

This is a quiz, so no re-submission allowed. Make sure to verify code and zip file before uploading!

You are required to create a diet tracking app (console app in Java). The app allows the user to keep track of all items consumed and how many calories they have eaten based on how much of all the four food groups they have eaten. Each food group has a different computation for how many calories per 100g, as follows:

1) Fruits and Vegetables – 65 calories per 100g
2) Meats and Proteins – 143 calories per 100g
3) Dairy – 42 calories per 100g
4) Grains – 265 calories per 100g

The user enters food items they have consumed into the app. A consumed food item consists of:

- Food Group: Food group of item
- Amount: Number of grams consumed of that item. Must not be negative.

The user has to provide both inputs to add the information to the diet tracking app.

Requirements

1. Create a set of classes to represent the different types of food items.
    a. Hint: abstraction/inheritance
2. Using polymorphism, each food item should have a method getCaloriesConsumed() that computes the amount of calories it represents based on its food group and grams consumed
    a. Note: It is possible to do this without polymorphism, but you must use polymorphism here for full marks.
3. Create a method getFoodItemConsumed() to get a food item from the user via the console
    a. Hint: switch statement is probably needed
4. The user should be able to enter multiple food items
    a. Hint: Loop
5. Store all food items entered by the user in a List
    a. Hint: `List<FoodItem>`
6. Create a method computeAllCaloriesConsumed() that computes the total calories of all food items that have been entered by the user and stored in the list.
7. When the user is done entering all food items, print the total calories consumed to the console

Note: You do NOT have to use MVC design.

Practical Quiz Directions

- Submit zip of program files to Lea. (Zip of main/java folder is sufficient).
- Mark will be based on Functionality, Correctness and Design
- Functionality (penalty-based grading)
    - If code does not compile, a failing mark will be given for the practical portion, possibly 0.
    - If code has runtime errors, appropriate marks will be lost based on severity of error on functionality
    - MAKE SURE TO TEST BEFORE SUBMITTING
- Design (8 points)
    - Apply appropriate OOP design (monolithic design receives lower marks). i.e., use appropriate abstraction, encapsulation, inheritance and polymorphism.
- Correctness and Completeness (16 points)
    - Goal: All features implemented as specified. Marking based on how well each feature is implemented and how many features are implemented.
    - Some features may be less important than others (as indicated).
    - Use comments in code if there is something you couldn't get working – Explain what you were trying to accomplish. Partial marks may be awarded.