Programming Exercise 01 - Basic Java

Authors: Sreya, Faris, Shreiyas, Akul

Problem Description

This assignment will test your basic knowledge of variables, expressions, assignment, classes, and String output.

There are two parts:

1. Currency conversion
2. Calculate how much battery is left on your phone

Solution Description

Part 1: Currency conversion

1. Create a class called `CurrencyConversion`
2. Add a comment (block or single-line) in your class, including your name and a fun fact about you.
3. Create the main method inside the `CurrencyConversion` class – all next steps will be inside the main method.
4. Create a variable named `aedPerUsd` and assign to it a value of 3.67.
   - The type of this variable should be the default type used by Java for floating-point values HINT: think about the primitive literals
5. Create a variable named `aedPerEur` and assign to it a value of 4.19.
   - The type of this variable should be the default type used by Java for floating-point values HINT: think about the primitive literals
6. Create a variable named `notes` and assign to it a value of 250.
   - The type of this variable should be the default type used by Java for integer values HINT: think about the primitive literals
7. Create a `String` named `name` and assign to it your name (e.g. "Faris").
8. Calculate how many `euros` and `dollars` you have and put your results in two new variables using the following formulas. These variables should be of the same type as `aedPerEur` and `aedPerUsd`:
   - `euros = notes / aedPerEur`
   - `dollars = notes / aedPerUsd`
9. Finally, use this exact statement to print how many Euros and Dollars you are carrying (We'll cover using `printf` in a future lecture.):

```
System.out.printf(name + " is carrying %.2f Euros and %.2f Dollars!",
euros, dollars);
```

10. Your output should look like this, but with your first name at the beginning of the line:

```
Faris is carrying 59.67 Euros and 68.12 Dollars!
```

Part 2: Calculate how much battery is left on your phone

1. Create a class called `BatteryCalculator`
2. Create the main method inside the `BatteryCalculator` class – all next steps will be inside the main method
3. Create a variable named `tikTokHours` and assign to it a value of 2.
    a. The type of this variable should be the default type used by Java for integer values
    HINT: think about the primitive literals
4. Create a variable named `spotifyHours` and assign to it a value of 3.
    a. The type of this variable should be the default type used by Java for integer values
    HINT: think about the primitive literals.
5. Create a variable named `messagingHours` and assign to it a value of 1.5.
    a. The type of this variable should be the default type used by Java for floating-point values
    HINT: think about the primitive literals
6. Create a variable named `instagramHours` and assign to it a value of 0.5.
    a. The type of this variable should be the default type used by Java for floating-point values
    HINT: think about the primitive literals
7. Create a variable named `phoneBattery` and assign to it a value of 100.
    a. The type of this variable should be the default type used by Java for integer values
    HINT: think about the primitive literals
8. **Using Compound Assignment operators (+=, -=, etc)**, perform the following calculations to `phoneBattery`
    a. Subtract `tikTokHours` multiplied by 17 (meaning each hour drains your battery by 17%)
    b. Subtract `spotifyHours` multiplied by 5 (meaning each hour drains your battery by 5%)
    c. Subtract `messagingHours` multiplied by 8 (meaning each hour drains your battery by 8%)
    d. Subtract `instagramHours` multiplied by 12 (meaning each hour drains your battery by 12%)
    e. Note that the values of `tikTokHours, spotifyHours`, etc. should not change
9. Finally, use this exact statement into your code to print out the final value of `phoneBattery`. (We'll cover using printf in a future lecture.)

```
System.out.printf("The battery of my phone at the end of the day is
%d%%. \n", phoneBattery);
```

10. Here is what the output should look like with the given numbers:

```
The battery of my phone at the end of the day is 33%.
```

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our auto grader. For that reason, do not use any of the following in your final submission:

- `var` (the reserved keyword)
- `System.exit`

Import Restrictions

You may not import anything for this homework assignment.

Collaboration

Only discussion of the PE at a conceptual high level is allowed. You can discuss course concepts and HW assignments broadly, that is, at a conceptual level to increase your understanding. If you find yourself dropping to a level where specific Java code is being discussed, that is going too far. Those discussions should be reserved for the instructor and TAs. To be clear, you should never exchange code related to an assignment with anyone other than the instructor and TAs.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files.
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Ed Discussion for all official clarifications

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- `CurrencyConversion.java`
- `BatteryCalculator.java`

Make sure you see the message stating "PE01 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section. **Any autograder test are provided as a courtesy to help "sanity test" your work and you may not see all the test cases used to grade your work.** You are responsible for thoroughly testing your submission on your own to ensure you have fulfilled the requirements of this assignment. If you have questions about the requirements given, reach out to a TA or professor via Piazza for clarification.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the assignment (submit early and often). We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

1. Prevent upload mistakes (e.g. forgetting checkstyle, non-compiling code)
2. Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine. (We are not using checkstyle for this programming exercise, but in future HWs we will).

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.