CPSC-121
Lab 02: Time Conversion

Lab Description
You will write two applications that deal with keeping and displaying time in a clock, computer, phone or some other device. In computing, time is typically represented as a number of seconds that have passed since a fixed point in the past.

Since time is represented as a number of seconds, we must convert the number of seconds to hours, minutes and seconds before displaying it to the user. Similarly, if a user sets the time in hours, minutes and seconds, we have to convert it to seconds before storing it in the device.

Objectives
1. Write Java applications that use basic data types, variables, constants, and arithmetic expressions.
2. Gain experience using integer division to our advantage as well as the modulo (%) operator.
3. Practice using the Scanner object to read in user input from the keyboard.
4. Gain experience in creating applications and writing Java classes in VS Code.

Part 1: Writing Applications using Variables and Expressions

1.1: Convert hours, minutes, seconds into seconds

First, you will write a program that converts hours, minutes, and seconds into a total number of seconds.

1. Create a new Java program called **ConvertToSeconds**.
2. Complete the **ConvertToSeconds** class so that it does the following:
    a) Declares 3 integer variables which represent hours, minutes, and seconds. You may assume that the values entered are greater than or equal to zero.
    b) Prompts the user to enter the appropriate values.
    c) Assign the values entered by the user on the keyboard to each variable.
    d) Calculate and print the equivalent number of seconds.
    e) Prints the result to the user.

Here is a test case with 1 hour, 28 minutes, and 42 seconds -
Output:

```
Hours: 1
Minutes: 28
Seconds: 42
Total seconds: 5322
```

1.2: Convert total seconds into hours, minutes and seconds

Now, you will write another program that does the reverse computation.

1. Create a new Java program named **ConvertToHours**.
2. Complete the **ConvertToHours** class so it does the following:
    a) Declare an integer variable which represents the total number of seconds.
    b) Prompt the user to enter the appropriate value.
    c) Assign the value entered by the user on the keyboard to the variable.
    d) Calculate the equivalent amount of time as a combination of hours, minutes and seconds; integer division and the modulo operator (%) will be needed here.
    e) Print the results to the console.

Here is a test case with 5,322 seconds -
Output:

```
Total seconds: 5322
Hours: 1
Minutes: 28
Seconds: 42
```

1.3: Convert seconds into hours (fractional)

Next, you will add logic to your **ConvertToHours** program so that it will also convert the given number of seconds to a fractional number of hours.

1. Open your file named **ConvertToHours.java**.
2. Add code to your existing program so it does the following:
    a) Calculate the entered number of seconds to a fractional number of hours; in-line conversion will be needed here
    b) Print the fractional result to the console.

Here is a test case with 5,322 seconds -
Output:

```
Total seconds: 5322
Hours: 1
Minutes: 28
Seconds: 42

Fractional hours: 1.4783333333333333
```

1.4: README

Update the plain text file called README and write your name and class section on the top. Follow the directions as described in README; answering the questions as you remove the question text.

Documentation: README

Update the plain text file called README and write your name and class section on the top.

1. Fill out the **Overview** section
   Concisely explain what the program does. If this exceeds a couple of sentences, you are going too far. I do not want you to just cut and paste, but paraphrase what is stated in the project specification.
2. Fill out the **Compiling and Using** section
   This section should tell the user how to compile and run your code. It is also appropriate to instruct the user how to use your code. Does your program require user input? If so, what does your user need to know about it to use it as quickly as possible?
3. Fill out the **Discussion** section
   Think about and answer the following reflection questions as completely as you can. You may not have an "earth-shattering" reflection response to each one but you should be as thoughtful as you can.

   Reflections...
   - What problems did you have? What went well?
   - What process did you go through to create the program?
   - What did you have to research and learn that was new?
   - What kinds of errors did you get? How did you fix them?
   - What parts of the project did you find challenging?
   - Is there anything that finally "clicked" for you in the process of working on this code?
   - Is there anything that you would change about the lab?
   - Can you apply what you learned in this lab to future projects?
4. Fill out the **Testing** section
   You are expected to test your projects before submitting them for final grading. Pretend that your instructor is your manager or customer at work. How did you ensure that you are delivering a working solution to the requirements?

Rubric

- Does it work as described? (Needs my sign-off): **10 points**
- Did you follow the directions? (Including well-thought out README file): **5 points**
- Did you follow style and coding practices: **10 points**
- **Total: 25 points**

Submitting the Lab

Compress the submission files (see below) into a .zip file named: **Lab02_*LastName_FirstName*.zip** .
When you are done and ready to submit, submit the following files in the .zip file:

- `ConvertToSeconds.java`
- `ConvertToHours.java`
- `README.txt`