# Programming Exercise 01 – Basic Java

Authors: Ruchi, Ricky, Daniel

Topics: variables, assignments, expressions, comments

## Problem Description

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

There are two parts:

1. Calculate your change after taxes (`ShoppingMall.java`).
2. Calculate the time (`TimeCalculator.java`).

## Solution Description

### *Part 1: ShoppingMall*

1. Create a class called `ShoppingMall`.
2. Add a comment (block or single-line) in your class and write your name and a fun fact about you.
3. Create the `main` method inside the `ShoppingMall` class – all the next steps will be inside the `main` method.
4. Create a variable named `cash` and assign to it a value of 100.
    - The type of this variable should be the default type used for integer literals.
5. Create a variable named `taxRate` and assign to it a value of 1.13.
    - The type of this variable should be the default type used for floating-point literals.
6. Create a variable named `subtotal` and assign it a value of 58.62.
    - The type of this variable should be the default type used for floating-point literals.
7. Create a String variable named `name` and assign to it your name.
8. Store the amount of cash remaining after taxes into a variable of appropriate type named `change`, calculated using the following formula:
    - `change = cash – subtotal * taxRate`
9. Create a variable named `changeTrunc` of appropriate type for the following calculation. Using the `change` variable calculated above, "truncate" the decimal portion to **two decimal places**. You must use basic arithmetic operators (*, /) and casting to accomplish this. Keep in mind that casting is NOT rounding (e.g., 3.519 truncated to two decimal places is 3.51).
    - **Hint:** Think about how we can "move" a decimal point left or right, and how we can remove the digits following the decimal point.
10. Finally, use the following statement (type it out, do not copy and paste) to print how fast you were driving:

```
System.out.println(name + " has $" + changeTrunc + " dollars
remaining!\n" + name + " started with $" + cash + "
dollars!");
```

The output should be as follows, but with your name at the beginning of the line:

```
Daniel has $33.75 dollars remaining!
Daniel started with $100 dollars!
```

## *Part 2: Time Calculator*

1. Create a class called `TimeCalculator`.
2. Create the `main` method inside the `TimeCalculator` class – all the next steps will be inside the `main` method.
3. Create a variable named `homeworkTime` and assign to it a value of 3.99.
    - The type of this variable should be the default type used for floating-point literals.
4. Create a variable named `lectureTime` and assign to it a value of 1.25.
    - The type of this variable should be the default type used for floating-point literals.
5. Create a variable named `numLectures` and assign to it a value of 4.
    - The type of this variable should be the default type used for integer literals.
6. Create a variable named `breakTime` and assign to it a value of 0.53.
    - The type of this variable should be the default type used for floating-point literals.
7. Create a variable named `travelAdjustment` and assign to it a value of 0.08.
    - The type of this variable should be the default type used for floating-point literals.
8. Create a variable named `timeTotal` and assign to it a value of 0.0.
    - The type of this variable should be the default type used for floating-point literals.
9. Using compound assignment operators (+=, -=, etc.), perform the following calculations to `timeTotal`. Use one compound assignment operator per statement.
    a. Add `homeworkTime` divided by 2 (your homework took half as long as expected!).
    b. Add `lectureTime` multiplied by `numLectures` (you go to `numLectures` classes).
    c. Subtract `breakTime` multiplied by 3 (you take three study breaks).
    d. Multiply by `1 + travelAdjustment` (you have to move to and from classes).
10. Create a variable named `timeTotalTrunc` of appropriate type and assign to it the value of `timeTotal` truncated to two decimal places. You must use basic arithmetic operators (*, /) and casting to accomplish this.
11. Use the following statement (type it out, do not copy and paste) to print the total cost of the order:

```
System.out.println("You're busy for " + timeTotalTrunc + " hours.");
```

12. Create a variable named `totalHours` and assign to it a value of 24.
    - The type of this variable should be the default type used for integer literals.
13. Using a compound assignment operator, subtract `timeTotalTrunc` from `totalHours`.

14. Use the following statement (type it out, do not copy and paste) to print out the dollars you have remaining:

```
System.out.println("You have " + totalHours + " hours
remaining in the day.");
```

15. With the given numbers, the output should look like the following:

```
You're busy for 5.83 hours.
You have 18 hours remaining in the day.
```

## Turn-In Procedure

### *Submission*

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

- `ShoppingMall.java`
- `TimeCalculator.java`

Make sure you see the message stating the assignment was submitted successfully. From this point, Gradescope will run a basic autograder on your submission as discussed in the next section. **Any autograder tests are provided as a courtesy to help "sanity check" 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 the class forum 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. We will only grade your latest submission. **Be sure to submit every file each time you resubmit**.

### *Gradescope Autograder*

If an autograder is enabled for this assignment, you may be able to see the results of a few basic test cases on your code. Typically, tests will correspond to a rubric item, and the score returned represents the performance of your code on those rubric items only. 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:

- Prevent upload mistakes (e.g., non-compiling code)
- 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.

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.

## Allowed Imports

To prevent trivialization of the assignment, you may not import any classes for this assignment.

## Feature Restrictions

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

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

## Collaboration

Only discussion of the assignment at a conceptual high level is allowed. You can discuss course concepts and 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 a note containing all official clarifications and sample outputs.**

It is expected that everyone will follow the Student-Faculty Expectations document and the Student Code of Conduct. The professor expects a **positive, respectful, and engaged academic environment** inside the classroom, outside the classroom, in all electronic communications, on all file submissions, and on any document submitted throughout the duration of the course. **No inappropriate language is to be used, and any assignment deemed by the professor to contain inappropriate offensive language or threats will get a zero.** You are to use professionalism in your work. Violations of this conduct policy will be turned over to the Office of Student Integrity for misconduct.