Tutorial 8: Enums, Switch Expressions, Sealed Classes, and Records

CSE1100 - Object Oriented Programming

1 Enums and Switch

We have created a `BankingApplication`. We can make some improvements to this application to make it more easy to read. One idea is to store all possible input options in an `enum`, so we can read `case SHOW_BALANCE` instead of `case 1` (and the same for the other cases). A second improvement we can make is to change the switch statement to a switch expression.

- Create an `enum` that represents the different options of the application.
- Change the type of `option` to this enum.
- Change the switch statement in `executeOption` to a switch *expression*.

2 Sealed and Records

Given is a small calculator application. In the `expression` package, you can find one `Expression` interface and four implementations of that interface: `Constant`, `Add`, `Subtract`, and `Multiply`. These represent constant numbers (e.g. '1', '-5'), addition (e.g. '1 + 2'), subtraction (e.g. '4 - 2'), and multiplication (e.g. '4 * 2') respectively. The `Calculator` class contains one method `calculate` that will calculate the value of any expression. This application works perfectly fine, but it could still use some improvements.

2.1 Records

If you look closely, you will see that the classes `Constant`, `Add`, `Subtract`, and `Multiply` only store some constant data. We can make the implementation of those classes a lot more concise by converting them to `record`s. Convert these classes to `record`s.

2.2 Sealed

The last line of the `calculate` function should ideally never be executed. We can enforce this by making the `Expression` interface sealed. Seal the `Expression` interface and make it permit `Constant`, `Add`, `Subtract`, and `Multiply` as subclasses.

Although, the last line of the `calculate` method should now not be called, we unfortunately cannot remove it until Java 21.

2.3 Instanceof Pattern

One final improvement we can make is to remove the casts from the `calculate` function. Whenever we do `x instanceof C` and then a cast `(C) x`, we can replace this by `x instanceof C y`, where `y` is an instance of class `C`. Remove all the casts from the calculate function.