The purpose of this set of optional exercises is to review some of the concepts seen so far during the course.

**Exercise 1 – Generics (Optional recap)**

Open the es1 folder in your IDE. Inside, you'll find a class hierarchy representing various types of beers and a *List* containing different beer objects.

The goal of this exercise is to implement the following methods:

- Create an *add()* method that requires the following 3 parameters: a list of beers called *source*, a *Class* representing the beer *type*, and a collection of *Beer* called *target.* Populate the *target* collection with all beers from the *source* that match the specified beer *type*. Use **bounded wildcards** for both the *source* and *target* collections, as well as for the *type*.
- Develop a *filter()* method that takes a list of beers and a *Class* parameter representing the beer type and returns a new list of beers containing only beers of the specified class type.
- Implement an alternative version of the previous filter method, by evolving it into a **generic** method called *filterGeneric()*. The type of the returned list has to match with the beer type specified as parameter.

To clarify, consider the following method calls:

```
List<Beer> beers …;
add(beers, Weisse.class, addedWeisse);
List<Beer> trappists   = filter(beers, Trappist.class);
List<PaleAle> paleAles = filterGeneric(beers, PaleAle.class);
```

**Hint**: use Java's reflection to verify if an element of the list is of the same type as the beer type specified as parameter.

**Exercise 2 – Annotation (Optional recap)**

A changelog is a document that provides information about the changes, upgrades, and enhancements made to software. It usually comprises details such as the version number, a chronological list of newly added features, bug fixes, improvements, and other changes.

The goal of this exercise is to create the *ChangeLog* annotation for documenting changes to classes, interfaces, fields, methods, and constructors. This annotation should allow to specify the type of change (ADDED, FIXED, CHANGED, REMOVED). It should also permit the inclusion of a brief description of the change and the specification of the version in which the change has been made. Furthermore, it must be possible to add multiple annotations, for example, on a method, to track all major changes across versions.

Develop the code to generate a report that organizes all the changes based on their version (in ascending order) and their type.

```
# Changelog for es2.Coordinate

## Version 1.0.0
### Added
 - first implementation [es2.Coordinate]

## Version 1.0.1
### Fixed
 - Wrong approximation [es2.Coordinate.distance]

## Version 1.0.2
### Changed
 - added changed precision from double to float [es2.Coordinate.lat]
 - added changed precision from double to float [es2.Coordinate.lon]
### Removed
 - compareTo implementation [es2.Coordinate]
```

```
## Version 1.0.3
### Changed
 – added final [es2.Coordinate.lat]
 – added final [es2.Coordinate.lon]
```

**Exercise 3 – Streams (Optional recap)**

Open the source code es3/Exercise3.java in your IDE. It provides an example of how to create a *Stream* of type *Character* from a *String*.

The goal of this exercise is to use Java's Streams API to:

- Determine the count of unique characters.
- Find the number of letters and non-letters.
- Check if there are any numbers (digits).
- Count the occurrence of each Character, with the results sorted by the character itself.
- Identify the ten most common letters.
- Discover the most common vocal.