

- Mark your completed exercises in the OLAT course of the PS.
- You can start from `template_07.hs` provided on the proseminar page.
- Your .hs-file(s) should be compilable with ghci and be uploaded in OLAT.

Exercise 1 *Rational Numbers***5 p.**

Implement rational numbers in Haskell. Here, rational numbers are represented by two integers, the numerator and the denominator. For instance the rational number $-\frac{3}{5}$ is represented as `Rat (-3) 5` when using the following data type definition.

```
data Rat = Rat Integer Integer
```

1. Implement a normalisation function for rational numbers, so that all of `Rat 2 4`, `Rat (-1) (-2)` and `Rat 1 2` get transformed into the same internal representation. (1 point)

Hint: the Prelude already contains a function `gcd`.

2. Make `Rat` an instance of `Eq` and `Ord`. Of course, `Rat 2 4 == Rat 1 2` should be valid. (1 point)
3. Make `Rat` an instance of `Show`. Make sure that `show r1 == show r2` whenever `r1 == r2` for two rational numbers `r1` and `r2`. In particular, `show (Rat 1 2) == show (Rat 2 4)` should evaluate to true. Moreover, integers should be represented without division operator. (1 point)

Examples: `show (Rat (-4) (-1)) == "4"` and both `show (Rat (-3) 2)` and `show (Rat 3 (-2))` result in `"-3/2"`.

4. Make `Rat` an instance of `Num`. See <https://hackage.haskell.org/package/base-4.16.0.0/docs/Prelude.html#t:Num> for a detailed description of this type class. (2 points)

Exercise 2 *Class and Data***5 p.**

1. Create a new data type `Ingredient`, which should hold the name, the amount and the unit of the ingredient. For the unit there exists ML (milliliters), G (gram) and PC (pieces). Also make `Ingredient` an instance of `Show` to show an ingredient like this `"200 ML of Milk"`. (1 point)
2. Create a new `class Price` with a function `getPrice`, which should return a price in euro as a `Float`. Create now an instance for your `Ingredient` data type. The price of an ingredient is defined independently of the name of the ingredient as follows: 1 ML costs 0.12 cent, 1 PC costs 75 cent and 1 G costs 0.095 cent.

Modify the `Show` instance of `Ingredient` of the previous part in a way that additionally the price of the ingredient is displayed in Euro, e.g., `"200 ML of Milk, cost: 0.24 EUR"` (2 points)

3. Create a data type `Recipe`, where you can store an arbitrary amount of ingredients. Make `Recipe` an instance of `Price` which calculates the total costs of the ingredients. Moreover, make `Recipe` an instance of `Show` such that a recipe is displayed in a form like this: `"200.0 ML of Milk, cost: 0.24 EUR - 200.0 G of Sugar, cost: 0.19 EUR - 3.0 PC of Egg, cost: 2.25 EUR - Price of the Recipe: 2.68 EUR"`. (2 points)