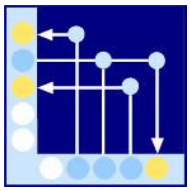
Fakultät Elektronik und Informatik
Studienbereich Informatik



Advanced Programming with MOSTflexiPL

Lecture in Wintersemester 2025/2026
Prof. Dr. habil. Christian Heinlein

8. Task Sheet (December 23, 2025)

Task 14: Rational Numbers

Subtask 14.a)

Define a type `rat` for the representation of rational numbers, i. e., fractions with integral numerator and denominator.

A *natural* fraction is a `rat` object whose numerator and denominator are natural `int` values and whose denominator is different from 0 (i. e., the integer division of the numerator by the denominator does not return nil). All other `rat` objects including nil are *unnatural* fractions.

A *normalized* fraction is a natural fraction which is completely canceled (i. e., numerator and denominator do not have any common divisor) and whose denominator is positive (i. e., a negative sign is only allowed in the numerator).

Subtask 14.b)

Define the following operators:

- For `int` values `n` and `d`, `n/d` returns a fraction with numerator `n` and denominator `d`.
- For a `rat` value `r`, `-r` returns the fraction `r` with inverse sign.
- For `rat` values `r1` and `r2` and `int` values `i1` and `i2`, `r1+r2`, `r1+i2`, and `i1+r2` return the sum of `r1` or `i1`, respectively, and `r2` or `i2`, respectively, as a fraction. (However, `i1+i2` must still return the sum of `i1` and `i2` as an `int` value.)
- Accordingly for difference (`r1-r2`, `r1-i2`, `i1-r2`), product (`r1*r2`, `r1*i2`, `i1*r2`), and quotient (`r1/r2`, `r1/i2`, `i1/r2`).

All operators mentioned above always return either normalized fractions (if this is possible) or nil (if the result would be an unnatural fraction).

Each of the operators mentioned above shall have the same binding properties as the corresponding predefined operators for `int` values, where `•/•` corresponds to the predefined operator `•:•`.

Subtask 14.c)

Define, in analogy to task 7, an operator that allows chained comparisons of all kinds with two or more operands, each of which has either type `rat` or `int`, for example:

```
r1 := 3/4;
r2 := 5/6;

r1 < 1;
r1 < 1 > r2
```

Define, if necessary, in analogy to task 11.a, additional "intersection operators" in order to make all comparisons of `rat` and `int` values unambiguous, even if the comparison operators from task 7 and 8.b are also visible and/or a comparison would also match the predefined equality test, for example:

```
r1 = r2;
r1 /= r2
```

All operators defined in this subtask shall have the same binding properties as the predefined equality test.