Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecturer: Prof. B. Gärtner

Informatik für Mathematiker und Physiker HS14

Exercise Sheet 11

Submission deadline: 15:15 - Tuesday 2nd December, 2014
`Course URL:` http://lec.inf.ethz.ch/ifmp/2014/

Assignment 1 – Skript-Aufgabe 146 (4 points)

We want to have a function that *normalizes* a rational number, i.e. transforms it into the unique representation in which numerator and denominator are relatively prime, and the denominator is positive. For example,

$$\frac{21}{-14}$$

is normalized to

$$\frac{-3}{2}.$$

There are two natural versions of this function:

```
// POST: r is normalized
void normalize (rational& r);

// POST: return value is the normalization of r
rational normalize (const rational& r);
```

Implement one of them, and argue why you have chosen it over the other one.

Hint: you may want to use the function `gcd` from lecture 10, modified for arguments of type `int` (how does this modification look like?).

Assignment 2 – Skript-Aufgabe 145 (4 points)

a) Provide definitions for the following binary relational operators on the type `rational`. In doing this, try to reuse operators that are already defined.

```
// POST: return value is true if and only if a != b
bool operator!= (rational a, rational b);

// POST: return value is true if and only if a < b
bool operator< (rational a, rational b);

// POST: return value is true if and only if a <= b
bool operator<= (rational a, rational b);

// POST: return value is true if and only if a > b
bool operator> (rational a, rational b);
```

```
// POST: return value is true if and only if a >= b
bool operator>= (rational a, rational b);
```

On the website you find `rational.cpp` and `rational_test.cpp`. The file `rational.cpp` defines the struct `rational` and contains some of the operators that were discussed in the lecture. Add your solutions to this file. Then compile and run `rational_test.cpp` to test your functions on some examples.

b) Write a program `rational_sort.cpp` that reads an unknown number of rational numbers from standard input into a vector by using `push_back` (as done in Sheet 9, Exercise 2). After reading the values, the program should sort them and output the sorted sequence. For example on input `6/7 1/2 1/4` the output should be `1/4 1/2 6/7`. You also find some larger sample inputs on the website.

   Hint: Use the operators that you have defined for the type `rational` or that were already part of `rational.cpp` before. If `operator<` is defined, `std::sort` can be used to sort the vector!

Assignment 3 (4 points)

In this exercise we will implement operations for 2×2 matrices with elements of type `double`. Again try to reuse as much code as possible. Furthermore, there is a template `matrix_operations_template.cpp` on the website. You may use it if you want.

a) Implement a type `Matrix` to represent 2×2 matrices with entries of type double.

b) Implement the following matrix operations for your type: matrix addition, matrix multiplication, and scalar multiplication for matrices.

c) Implement `operator>>` and `operator<<` for your `Matrix` type.

d) Implement a `main` function for your program which reads two matrices `A` and `B`, and a factor `c` of type `double` from the user. Your program should then output `A+B`, `A*B`, `c*A`, and `B*c`.

Challenge – Skript-Aufgabe 149 (8 points)

Hint: The `window` library `libwindow` is already pre-installed on your VirtualBox. To use it you have to `#include <IFMP/window>` In the VirtualBox you can also find two small demo programs `demo1.cpp` and `demo2.cpp`, which show you how to use the `window` library, in the folder

    ~/Desktop/progs/libraries/libwindow/demo/