Assignment 6 - Debugging and Unit Tests

- The problems of this assignment must be solved in C++.
- The TAs are grading solutions to the problems according to the following criteria: https://grader.eecs.jacobs-university.de/courses/320143/2018_1r2/Grading-Criteria-C++.pdf

Problem 6.1 *Debug with `gdb` I* (1 point)

Presence assignment, due by 18:30 h today

Download

`https://grader.eecs.jacobs-university.de/courses/320142/cpp/debug1.cpp`

Use `gdb` or some other debugger to determine the reasons of the wrong results. Describe within comments in your program the `gdb` commands and steps you used and how you found the reasons of the wrong results.

Problem 6.2 *Debug with `gdb` II* (2 points)

Presence assignment, due by 18:30 h today

Download

`https://grader.eecs.jacobs-university.de/courses/320142/cpp/debug2.cpp`

Use `gdb` or some other debugger to determine the reasons of the wrong results and/or the crashing during the program execution. Describe within comments in your program the `gdb` commands and steps you used and how you found the reasons of the wrong results and/or crashing.

Problem 6.3 *Simple unit tests* (2 points)

Write a `Fraction` class for working with rational numbers. Make sure that you have two constructors, one with two parameters, two integers and another one with a string as a parameter containing the information about the rational number (e.g., "`-7/89`", "`-9/-98`", "`+1/-9`", etc.). Overload all relational operators: `<`, `<=`, `>`, `>=`, `==`, `!=`, then the arithmetic operators: `+`, `-`, `*`, `/`, then the I/O operators `>>` (the values should be separated by space or newline), `<<` (the values should be separated by `/` and ended by newline, e.g., `-1/2`), and then the assignment operator `=`. Also write a copy constructor. Write a `main` function which contains tests for all the above operators, constructor calls and exception throws. Write the implementations in a way that in case of invalid data or logical errors the methods or functions throw a string exception with the message "`Invalid data or logical error`".

Submit all source files as a zip archive.

Problem 6.4 *Classify unit tests with Makefile* (3 points)

Continue with your previous implementation by splitting the different tests into multiple `cpp` files: testing the relational operators, testing the arithmetic operators, testing the I/O operators, and testing the exceptions (constructors and the assignment operator). Write a makefile for the classification of the tests such that you can compile and run all test groups separately as well as compile and run them all together.

Submit all source files and your makefile as a zip archive.

How to submit your solutions

- Your source code should be properly indented and compile with `g++` without any warnings (You can use `g++ -Wall -o program program.cpp`). Insert suitable comments (not on every line ...) to explain what your program does.
- Please name the programs according to the suggested filenames (they should match the description of the problem) in Grader.
  Each program **must** include a comment on the top like the following:

```
/*
   CH08-320143
   a6_p1.cpp
   Firstname Lastname
   myemail@jacobs-university.de
*/
```

- You have to submit your solutions via *Grader* at
  `https://grader.eecs.jacobs-university.de`.
  If there are problems (but **only** then) you can submit the programs by sending mail to `k.lipskoch@jacobs-university.de` **with a subject line that begins with CH08-320143**.
  It is important that you do begin your subject with the coursenumber, otherwise I might have problems to identify your submission.
- Please note, that after the deadline it will not be possible to submit any solutions. It is useless to send late solutions by mail, because they will not be accepted.

This assignment is due by Wednesday, March 21st, 10:00 h.