Programming in C++ II, Rotation 2
Jacobs University Bremen
Dr. Kinga Lipskoch

Course: CH08-320143
March 7th, 2019

Assignment 1 - Operator Overloading, Streams and Multiple Inheritance

- 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/320143/2019_1r2/Grading-Criteria-C++.pdf

Problem 1.1 *`Complex` class with streams* (2 points)

Presence assignment, due by 18:30 h today

Adapt and extend your previous source code for working with complex numbers (`Complex.h`, `Complex.cpp` and `testComplex.cpp`) such that the operators + for adding, − for subtracting, * for multiplying two complex numbers, = for assigning, << and >> for input and output streams are overloaded. Your testing program (`testComplex.cpp`) should read two complex numbers from the files called `in1.txt` and `in2.txt`, then compute their sum, the difference and the product, and print the results on the screen as well as into a file called `output.txt`. You have the freedom to set the structure of the input files.

You can assume that the input of the testing program will be valid and the necessary input files have a valid content if existing.

Problem 1.2 *Concatenate `n` files into new file* (2 points)

Write a program which reads from the standard input an integer value `n`, followed by `n` file names. Your program has to concatenate the content of the `n` files and write the result into the file called "`concatn.txt`" using binary read and write. Add a `'\n'` to separate the contents of the different files inside the resulting file. The operation of the concatenation is successful only if all files are existing and their opening was successful.

You can assume that the input will be valid and the necessary input files have a valid content if existing.

Problem 1.3 *Multiple inheritance I* (1 point)

Consider the following source file:
`https://grader.eecs.jacobs-university.de/courses/320142/cpp/minheritance1.cpp`
Compile and run the file. If you find any compilation errors, explain their reason as comments in the code. Then fix the errors such that the program compiles and runs. Explain the motivation of your modifications in the code and their effects on the execution.

Problem 1.4 *Multiple inheritance II* (1 point)

Consider the following source file:
`https://grader.eecs.jacobs-university.de/courses/320142/cpp/minheritance2.cpp`
Compile and run the file. If you find any compilation errors, explain their reason as comments in the code. Then fix the errors such that the program compiles and runs. Explain the motivation of your modifications in the code and their effects on the execution.

Bonus Problem 1.5 *`Matrix` and `Vector` classes* (3 points)

Extend your previous source code for working with vectors (`Vector.h` and `Vector.cpp`) and write a `Matrix` class (`Matrix.h`, `Matrix.cpp`) for operations with matrices. If you did not write a `Vector` class for one of the previous bonus problems, then write one with minimal functionality as needed for this problem. Then write a testing program which illustrates operations between matrices and vectors (`testmatvec.cpp`) using the operations described as in the following.

- Overload the operators << and >> to be able to enter matrices and vectors from the standard input and from files (e.g., `in1.txt`, `in2.txt`, etc.), and to send matrices and vectors to the standard output and to files (e.g., `out1.txt`, `out2.txt`, etc.). You have the freedom to set the structure of the input files.
- Overload the operator * for computing the product of a vector and a matrix, and a matrix and a vector. For simplicity reasons you can consider a vector to be either a row vector or a column vector such that mathematical multiplication with a matrix is possible. By this convention, the result of the multiplication is also a vector (row vector or column vector).

Your implementation has to check if the operations between matrix and vector, and vector and matrix are valid or not (concerning the compatibility between a vector and matrix concerning the amount of elements in them). Besides this you can assume that the input of the testing program will be valid, the necessary input files have a valid content if existing.

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
   a1_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 Tuesday, March 12th, 10:00 AM.