Programming Expertise
University of Potsdam SS-202X
Detlef Groth
Test - Exam - 02

You have 90 minutes time for the implementation of the programming tasks. Make after the steps 1-2 intermediate versions: FastaParser1.cpp, FastaParser2.cpp and FastaParser3.cpp. Submit at the end of the exam those single versions by USB stick or by E-Mail preferentially as zip archive (surname.zip) to me (E-Mail: dgroth@uni-potsdam.de). 75% of the tasks will be given on execute correctness and 25% of the tasks will be weighted by usefulness and clearness of the implementation.
The theory questions in 4 and 5 are to be answered first without any aids and the sheet with the answers is to be handed in after about 10-15 minutes. For the computer tasks 1-2 all aids are allowed during the exam. This does not include personal support from fellow students or other persons except from me.

Your regex reminder (you know what I mean ...):
^MA{1,2}[\\s\\t\\n].+.{1,}A.{0,}[^A-C]T.*A?A{0,1}*

Clarification

With this I state, that I will not take and give any not allowed support during the exam.

Name, Matrikel-Number.: Signature:

1. 2 points (layout console application)
2. 4 points (implementation of console application)
3. 3+1 points (advanced console application)

4+5. 4 points (C theory)

6. 2 points (C++ theory)

Sum: 12 points

Good luck !!

1. Layout of console application and utilizing command line arguments (2 points):
Create the basic outline of a console application with *main* and *help* function which explains the purpose of the application and the usage of command line arguments. Save the possible command line arguments in up to three variables. If the required arguments are not given, call the help function and exit the application. You can use just the argv array or an command line parser like popl or argparse. The arguments should be a option like `--search` or `--summary`, a filename for a FASTA sequence file and an optional sequence pattern for the `--search` option. Do a check if the given filename exists and warn the user if it does not exists on the users file system. Check if the options are either --help, --summary or –search. Provide an useful help message if the arguments are not correctly given using the help function. The help message should contain, program name, author name and an usage line like this as well as a short explanation of the arguments thereafter:

> *Detlef's FastaParser1 search FASTA files*
> *Author: Detlef Groth, University of Potsdam, 2024*
> *Usage: FastaParser1 --search|--summary|--help ?PATTERN? file.fasta*

Filename first task: _____

2. Opening and searching in the FASTA file (4 points): The input files we are working with are: sars-cov1.fasta and sars-cov2.fasta from the Moodle course. Our program should however work with any FASTA file. Implement a *searchSequence* function to display the sequence IDs which contain the given sequence search pattern. Hint: you should use regular expressions for searching in the file. The two arguments for this function should be the input filename and as second argument the search pattern. You should return the output for instance as a vector of tuples, giving the sequence ids, the pattern and if it hit does or does not matches the pattern and then print out the result in the main function. You can ignore the line break problem in your search. Below is a possible sample run of your program with parts of its tabular output:

$ FastaParser2 --search MAT sars-cov1.fasta
sp|P59637|VEMP_CVHSA MAT false
...
sp|P0C6X7|R1AB_CVHSA MAT ***true***

Filename second task: _____

3. Advanced Terminal Application (4 points):
1 (1 point): Extend your application so that as a C++ class and so that it can handle more than 1 filename at the same time given on the terminal. Hint: better handle multiple files in main, not within the function/method!

2) (3 points): Implement the summary function so that it prints out for all sequence ids the sequence lengths like this:

$ FastaParser3 --summary sars-cov1.fasta sars-cov2.fasta
sp|P0C6U8|R1A_CVHSA 4479
sp|P0C6X7|R1AB_CVHSA 7173
...
sp|P0DTC1|R1A_SARS2 4522
sp|P0DTC2|SPIKE_SARS2 1391
sp|P0DTC3|AP3A_SARS2 388 ...

Filename third task: _____