Advanced Software Engineering Lab (CS 480):

Lab 3: Design and build a software (website/standalone app/cell phone app/etc.) which can do basic arithmetic/trigonometric/logarithmic operations ("**+**","**-**","*****","**/**", "**^**", sin, cos, tan, cot, arcsin, arccos, arctan, arcctg, ln, $\log_{10}$) with real numbers involving parenthesis, brackets and curly brackets. The calculator software should evaluate properly the mathematic formula based on the precedence of the operators, the parenthesis and the aforementioned mathematical functions.

Instructions

- Use the programming language of your choice to build the calculator software and the corresponding interface. The interface can be a GUI or just a text based interface (see CLI). For details check the basic functionalities of a calculator. No external software library (except the math library if necessary) should be used to solve the problem!
- The "-" should be considered as unary and binary operator as well based on the corresponding syntax.
- The mathematic expression evaluation (see PN, RPN and shunting-yard algorithms if this is the choice you select) should be implemented directly in the software code. If some code snippet is considered from external sources, that should be explicitly mentioned in the source code.
- All the software changes/versions should be recorded in github or some other version control system.
- Create a gantt chart with all the tasks you accomplished during the software building including testing.

Submission guidelines

- A zip file containing only the source files + a readme file (HowTo_Firstname_Lastname.txt) describing how to compile/execute/interpret from command line the code packed into a Source_Code_LastName_Firstname.zip file. The instructor/TA should be able to compile/run/interpret the code from command line using the instructions from the readme file. For command line see in Windows command prompt or power shell, while in Linux/MacOS any regular shell of your choice.
- A gantt chart which records all the different stages the software development lifecycle. In order to avoid issues with different software packages creating the chart, please take a screenshot of the gantt chart and embed it in a pdf document. Name convention: Ghantt_Chart_Lastname_Firstname.pdf.
- A short document describing what type of version control system has been considered, what are the advantages and disadvantages of the versioning system in use, and a proof (see a screenshot for example embedded in the document) about the commits during the process. Name convention: Version_Control_Lastname_Firstname.pdf.
- If the software is not running/compiling/executing on the school machines or similar environments (considering regular compilers/interpreters such as Java, JavaScript, Python, C/C++ (GNU), Go, Rust, Matlab, etc. running on Windows, Linux or MacOS) or the instructor/TA is explicitly requesting, the software should be demoed "in-person" using the student's equipment and software setup via some video chat solution (see Zoom, Skype, etc.).

Grading criteria

1. The calculator should be able to correctly evaluate all kind of syntactically correct mathematic expressions. I.e. “-5.78+-(4-2.23)+sin(0)*cos(1)/(1+tan(2*-ln(-3+2*(1.23+arcsin(99.111))))=” (**4p.**)
2. The software should check for correct inputs. I.e. “**((**2+3/**)(**4**/-** = “ (**4p.**)
3. The gantt chart should contain all the phases considered by the developer involving the different software lifecycle stages. (**1p.**)
4. Usage of a version control system and proper description. (**1p.**)