CSE 411: Advanced Programming Methodology

Homework Assignment: Numerical Precision

Objective: This assignment gives you an opportunity to experience first-hand how numerical imprecision can corrupt the accuracy of mathematical calculations on computing systems. You will also see how you can trace and resolve these accuracy issues by computing things differently.

Background: Numerical accuracy is a major concern in a wide range of software from 3D gaming, simulation, image, speech, and pattern recognition, to bioinformatics, robotics, and financial software. The major concern is to eliminate the **creation** and **propagation** of numerical errors throughout the software, and to account for boundary cases.

Assignment: You are provided C++ code that will parse a text file containing a square matrix of order 3, a method that will print out a given matrix, and a method for matrix multiplication. The code also includes two skeleton methods for computing a matrix inverse. The main method runs both methods, thus computing the matrix inverse twice.

> **Task 1:** Fill in the first and second skeleton methods with a simple matrix inverse algorithm and a partial pivoting algorithm respectively.
> **Task 2:** Find a non-identity matrix with no zeroes that can be precisely inverted and is correctly inverted by both matrix inversion algorithms.
> **Task 3:** Find a non-identity matrix with no zeroes that cannot be precisely inverted by your simple inversion algorithm but can be inverted by your partial pivoting algorithm.
> **Task 4:** Explain why partial pivoting makes precise inversion possible in the previous task.

Compilation: The code is provided with a makefile for compiling your software. For grading purposes it is expected that you will not need to alter this make file, so do not do so – it will be used to compile your software.

Collaboration and online resources: You may discuss this assignment with other students in the class at a conceptual level only. Code may not be exchanged, and you are not permitted to view another student's code. Collaboration is not permitted. Also, you may not use any software available online. The code you write must be 100% your own.

Assignment Submission: Submit the entire directory tree as your assignment. Inside the debug directory, include files called "matrix1-<user>.txt" and "matrix2-<user>.txt" that contains the matrices you identified for Tasks 2 and 3. Include also a third file called "pivoting-<user>.txt" that provides your explanation for Task 4 above. The filenames should substitute the term "<user>" for your sunlab user name.

Evaluation: The software will be evaluated by first compiling it on sunlab, running it with the matrices you generated, and then running it with matrices other than those that you have generated. The grading rubric is:

60% for correctly operating code in Tasks 2, 3, 4.
10% no memory leaks, and robust on all inputs.
10% for good documentation, comments, compiles without warnings
5% for readable and concise code
5% best of reviewed group

You will receive zero points if your code does not compile. Code submitted in another language, **that uses a library, or that uses malloc or free,** will not be graded and will result in a zero. Use new and delete[]() only.

You will lose between 10 and 25 points if you have not commented every method in your code, in the beginning and throughout. Thorough and insightful comments are expected; grading will be rigorous. Use the makefile and provided code as a standard for high quality commenting.