Math 351, Fall 2011, Projects
Numerical Analysis I, Prof. Bownik

Instructions: 20% of your grade will come from a programming or research project due on the last day of classes, Friday, December 2. You can work individually or in teams of two. If working with another student, you should turn in a single project for both of you. You should decide on a topic within a week and then discuss with me either in person or by e-mail. I recommend that you turn in a rough draft before or immediately after Thanksgiving break in order to receive a feedback.

Computer programming: Topics can be written in any language, such as Java, C, C++, or Mathematica. You must submit your source code, which should be thoroughly commented, and also a text file with a description of the program.

Research project: Projects should be 4-5 typed pages and should include a discussion of the theory, examples of computations, and a complete bibliography. You might consider writing your research project as a Mathematica notebook, since it has built-in formula formatting and you can use it for your calculations.

Topics: You can choose any of "Student Research Projects" listed in the textbook, an assortment of computer problems from the textbook, or another project of your own design in consultation with me. Please choose a topic related to the material in Chapters 1–6. Most importantly, choose a topic that you find interesting. For example, a topic that ties Numerical Analysis to your other studies or interests. Here are a few possible topics for a project.

(1) **Computing π**. Investigate the algorithms for computing π. If doing this as a programming project, write a code to generate π up to at least 1000 decimal digits. If doing research project, describe several techniques for computing π, and compare and contrast them.
(2) **Polynomial interpolation demo**. Write a graphical demonstration of polynomial interpolation e.g. as a Java applet. The user should be able to drag data points in the plane and the program should compute interpolating polynomial and graph it in a real time. Hence, you will need to figure out the graphical interface.
(3) **Hybrid root approximation methods**. Investigate the hybrids of the bisection and the secant methods. Look up the references provided in Section 3.3 of the textbook to learn more about hybrid methods.
(4) **Muller's method**. Implement on a computer Muller's method for computing roots of equations. This algorithm is similar to the secant method. It uses polynomial interpolation with quadratic polynomials to determine recursively the next point in approximating sequence.
(5) **Newton's method for multiple variables**. Investigate the algorithms for finding roots of systems of equations.
(6) **Clenshaw-Curtis method**. Investigate this method for numerical integration which is often superior to the Romberg method.
(7) **Parallel numerical integration**. Investigate how numerical integration algorithms, which we have learned in this course, can be implemented on parallel processors and computers to speed up calculations.