Modeling and Simulation.

Practical Assignments 1 - finite precision.

For all of the assignments you are supposed to write a program in C, C++ or Python with suitable "general purpose" routines and a main routine to solve the particular problem in the assignment. Maintain a concise but comprehensive lab report to describe the results of your experiments with the code. The code should run on the student machines.
The assignments can be made in pairs. You do not need to make a full report, but make notes of your results and any issues that you encounter (maintain a lab report). Demonstrate your code to the assistant and be ready to explain your code and your results.
The central theme for this first set of assignments is the finite precision of number representation and the effects that can have on your results.

For reference you may want to visit
`http://en.wikipedia.org/wiki/Floating_point`
`http://www.tutorialspoint.com/c_standard_library/float_h.htm`
`http://www.gnuplot.info/`
`http://en.wikipedia.org/wiki/Numerical_differentiation`
`http://en.wikipedia.org/wiki/Root-finding_algorithm`

1 Floating point representation

a. Find out the size in bytes, the machine precision and the range of valid numbers for the machine you are using. In C you can find this information in float.h. Do this for floats, doubles and long doubles. Write a program to numerically check the machine precision and the range.

b. Write code to investigate the behaviour of your computer for invalid operations, NaN values, infinities and underflows (denormalised numbers). Make sure you cover a good number of cases. For NaN values you may want to look at the bit patterns.

c. Try to compute $\sum_{i=1}^{N} 1/i$ for $N = 10^8$ and $N = 210^8$ using floats and doubles in C. Use a forward summation approach and a reverse approach. Also try to come up with a strategy, also using floats, that yields a better / more accurate result. (C source code is actually provided, compile this with and without optimisation and try to explain the results. Computer Science and AI students must also implement the Kahan summation; for others this is optional.)

2 Numerical Differentiation

Write two routines to perform numerical differentiation on a given function, given a parameter value x and an increment h. Use right-hand and central differencing. Both functions will have three parameters: the function to be differentiated, x and h.
Now use these routines to find the derivative of $\sin(x)$ for $x = \pi/3, 100\pi + \pi/3, 10^{12}\pi + \pi/3$. Experiment with the value of h to find the most accurate result in each case: create a table and/or a graph showing the error as a function of h in each case.

3 Computing the root of 2

Calculate the value of $\sqrt{2}$ using the "false position" (regula falsi) method on the function $f(x) = x^2 - 2$. Record how many steps you need and at what rate the error decreases. Make a table or better a graph of $|f(x)|$ against step number. Should you use a linear or a logarithmic axis for the ordinate and/or the abscissa?
Now do the same experiment using Newton-Raphson. Again record how many steps you need and at what rate the error decreases.

4 Newton-Raphson

Use the Newton-Raphson method to compute the zeros of

$$f(x) = x^3 + 3x^2 - 4$$

How do you find a suitable starting value for x? Would it help to do the first few iterations using the bisection or regula falsi methods?
Can you determine the roots by hand?