Spring, 2017

Physics 6810: Assignment #1a

This assignment is designed to give you practice in the basic tasks from class, which we'll need repeatedly and build upon, and to verify to me that you can do them. Please ask questions! Because of our delayed start, the assignment has been split into two parts (1b will be handed out later.) This part is due by midnight on Friday, January 27 (and we'll iterate until done).

You can use either BuckeyeBox or Dropbox to "hand in" the assignment. If you didn't already do this in Activities 1, create a folder `6810_lastname_firstname` (e.g., `6810_Furnstahl_Dick` but use your name, not mine!) and share it with `furnstahl.1@osu.edu` and `smith.10851@osu.edu`. Then put your homework inside a subfolder named `PS_1a` or `Assignment1a` or similar.

Use C++ for the codes and gnuplot for the plots (see me if you want to use a different plotting program). *It is required that your code have appropriate comments.* Comment your codes with your name, email, AND revision history, as in the example codes from class. Check the 6810 webpage for suggestions and hints.

The overall goal of these tasks is to teach/remind you of some basic programming and reinforce some of the issues stemming from the approximate representation of real numbers.

1. **Update to `area.cpp`.**

   Implement items 1 to 6 from the "to do" list in the `area.cpp` code from Activities 1 in a new code called `area_new.cpp`. If C++ is new to you, don't panic! We'll iterate until it makes sense. If you are an experienced C++ program, do item 7 (implement a Circle class) instead.

2. **Summing up vs. summing down.**

   This problem is taken from problem 3 in section 3.4 of the Landau–Paez *Computational Physics* text, which examines the summation of $1/n$. The analysis should be similar to the one on finding the roots of quadratic equations from class (you might find the quadratic_equation_2.cpp listing useful). Consider the two series for integer N:

   $$S^{(\mathrm{up})} = \sum_{n=1}^{N} \frac{1}{n} \qquad S^{(\mathrm{down})} = \sum_{n=N}^{1} \frac{1}{n}$$

   Mathematically they are finite and equivalent, because it doesn't matter in what order you do the sum. However, when you sum numerically, $S^{(\mathrm{up})} \neq S^{(\mathrm{down})}$ because of round-off error.

(a) Write a program (and a makefile) to calculate $S^{(\mathrm{up})}$ and $S^{(\mathrm{down})}$ in single precision as functions of N. Make sure you include appropriate comments and indent it consistently.

(b) Make a log–log plot of the difference divided by the sum (or average), i.e.,

$$\frac{|S^{(\mathrm{up})} - S^{(\mathrm{down})}|}{\frac{1}{2}(|S^{(\mathrm{up})}| + |S^{(\mathrm{down})}|)} ,$$

versus N and turn in a postscript or pdf file of the plot.

(c) Interpret the different regions of your graph (e.g., Are the calculations equally precise in some regions or is one way of doing the calculation always better? Is there a region where the error looks like power law and if so, what power? [You don't need to *explain* the power, just extract the value.] What happens for large values of N?) Explain qualitatively in your own words why the downward sum is more precise. Put these discussions in the comments of your code at the top.