Abstract
Your goal for this project is to implement software emulation of floating point addition for 32-bit (single-precision) floating point numbers.

Input/Output
Your program will prompt the user for two floating point numbers. It will then compute and display the sum. Here’s example I/O from four runs (you only need to prompt once per execution run):

Enter a floating-point value: 1
Enter a floating-point value: 1
2.000000000000000000

Enter a floating-point value: 2.2
Enter a floating-point value: 1.4
3.599999904632568400

Enter a floating-point value: -1.34
Enter a floating-point value: 3.4
2.059999942779541000

Enter a floating-point value: 10.5
Enter a floating-point value: 100.2
110.6999694824219000

Issues to Resolve
Here’s a few issues:
- How will you deal with negative values?
- How will your normalizing algorithm work?
- What’s the easiest way to access bit fields within a word?

You may not use any floating-point instructions for this project!

There are I/O system calls for performing input and output of single precision floating point values. However, to use these system calls you will need to use the mfc1 and mtc1 instructions for copying words between the floating point registers and the integer (general-purpose) registers. More information about these instructions are available in the book (note that these are move instructions, not fp instructions).

You are not required to detect overflow or underflow, and you may always round down. However, for up to 20% extra credit, you can implement rounding (using the guard, round, and sticky bits).

My solution code was 71 lines long. Make sure you have a thorough understanding of floating point numbers and the addition algorithm before you start.

What to Submit
Submit your code via Dropbox.