Goals

- Investigate floating-point numbers and their properties
- Convert floating-point numbers from decimal to binary to normal form to register representations

General Instructions

IMPORTANT: when creating your document to print out and hand in, please use a fixed-width font such as Courier for your program!

The programs must be FULLY COMMENTED — header comments should provide information about author, date, program purpose (input/output and summary of processing performed); in-code comments should describe variables and summarize major steps in the algorithm. Use white space, e.g., blank lines, to make the program more readable. Use an indenting scheme (for instance, 4 space indents for function, loop, and if-statement bodies, nested appropriately). Use functions to simplify your code whenever it seems appropriate; be sure your functions also have header comments explaining their purpose, parameters, and return value (if any).

Part 1: Decode a Floating-Point Value Using Masks [25 points]

Write a C program that does the following:

- prompts the user for a floating-point value (float, not double)
- prints the 32-bit pattern of the value separated into three parts: sign, exponent, and significand (similar to the binary display in the Mars “Floating Point Representation” tool)
- prints the decimal representation of the adjusted exponent of 2 (subtract 127 from the binary value in the previous item)
- prints the decimal representation of the fraction plus the implicit leading 1
Here is a sample output—imitate this:

    ewire23-61:lab6 jwenskovitch$ ./decode
    Your Name
    Tue Oct 21 13:04:26 2013

Enter a float value: -11.1
In hex: c131999a
  1 10000010 01100011001100110011010
  Sign: -
  Exponent: 2^-3
  Fraction: 0.387500
  Significand: 1.387500

    ewire23-61:lab6 jwenskovitch$ ./decode
    Your Name
    Tue Oct 21 13:05:19 2013

Enter a float value: .005
In hex: 3ba3d70a
  0 01110111 01000111101011100001010
  Sign: +
  Exponent: 2^-8
  Fraction: 0.280000
  Significand: 1.280000

HINTS:

To obtain the binary pattern of the input float value, try using the “&” and “*” operators and casting.

Use unsigned rather than int.

You will need to write code to convert integer values into binary; if you can’t do this, ask me about it right away—don’t wait until next week!

To obtain the fractional part, you can just divide the integer value you get (from applying your mask) by the value $2^{23}$. 
Part 2: Floating-Point Number Conversion [15 points, 3 each]

Convert each of the following floating-point decimal values into their representations inside of a 32-bit register. For each, you must show step-by-step calculations converting the decimal values into binary values, converting the binary values into normal form, and converting the normal form into the registers themselves. You can use Mars to check your answers, but it should not give you the answer without associated work.

1. \((0.375)_{10} = (x)_2\)
2. \((1.375)_{10} = (x)_2\)
3. \((16.5625)_{10} = (x)_2\)
4. \((-7.25)_{10} = (x)_2\)
5. \((5)_{10} = (x)_2\)

Note that question #5 can be expressed as an integer as well, but the question is asking for the register representation if it is declared as a floating-point value:

```c
float x;
x = 5;
```