CS3650 Homework 3
Sept 26, 2019
Due Oct 3, 2019

For this homework, you will be creating either a Java class or a C++ class that implements floating point Add and Multiply. The layout of your class should look like this:

```
// Java Version
public class fp
{
        public String myName()
        {
                return "YourNameHere";
        }

        public int add(int a, int b)
        { … }

        public int mul(int a, int b)
        { … }
}
```

For the C++ version, the include file 'fp.h' declares the class, and the file 'fp.cpp' is a shell of the implementation. Take the fp.cpp file and enter your name and the bodies of the two functions.

Note that the routines take integer values and return an integer. However, these integers use the floating-point bit pattern. So, for example, I might pass 0x41C20000 as one of the inputs to your routines. This bit pattern on an integer corresponds to the floating point value 24.25. Inside the routine you will break this into the S, E, and F fields, then do the calculations for add or multiply, resulting in the S, E, and F fields of your response. Then put the S, E, and F back together to make an integer, which you return.

I also made a helper class, FPNumber, which helps with the process. In the constructor for FPNumber you pass the integer-that-has-a-floating-bit-pattern. It will split this into the S, E, and F fields. You can then ask for each of these values. At the end of the routine, you can also send in new values for S, E, and F, and the FPNumber will turn that back into an integer.

If you look in the Resources tab on Blackboard, you will see several files:

- fp.h
- fp.cpp
- fp.java
- FPNumber.java

If you are building the C++ version, take the first two. If you are building the Java version, take the second two.

Edit the fp.cpp file or the fp.java file, writing the implementations of the add and mul functions. When you are done, you can check these files into blackboard.

I have a test program that will call these routines with a number of test cases. Your grade for the homework will be based on the percentage of test cases that your code correctly solves.

As you can see, the inputs and outputs of the routines are all integers. However, the bits in these integers follows the IEEE754 standard, so bit 31 is the S field, bits 23 – 30 are the E field, and bits 0 – 22 are the F field.

Also, to help test your code, recall these numbers in FP format:

```
int v24_25 = 0x41C20000; // 24.25
int v_1875 = 0xBE400000; // -0.1875
int v5 = 0xC0A00000; // -5.0
```

Important notes:

1. Your code should not require any extra packages.
2. The book didn't adequately discuss rounding issues, so none of the test cases will require rounding.
3. The book did not accurately discuss denormalized numbers, so the only denormalized number in the test cases will be the value '0'.
4. But the examples will include zero, infinities, and some NaNs. In particular, figure out how to handle inputs that are infinity.