**CSC 325 Computer Architecture**

**Class 37 Floating Point Representation**

**November 25, 2013**

Submit your responses to Canvas by Monday, 12/2 before class.

Formal writing style required.

1. What is contained in the IEEE 784-2008 Standard report?
   1. This report contains information on floating point numbers and their pros as well as cons.
2. Summarize some of the errors that can occur when using the IEEE standard.
   1. One of the major errors that debuggers look for are NaN or Not a Number, which could cause an error unless the system is set up to handle NaNs, in which case there should be a switch for debuggers allowing and disabling NaNs.
   2. Debuggers should also be able to set patters or values for data not in use, for instance, using basic commands debuggers should be able to set all unused data to one or zero, or set it all in ascending order Ex. 1, 2, 3…
   3. It is also important to note that floats can cause error, owing to the fact that they are based on a calculation, if the system clock is not precise, small numbers can end up in a float, for instance 0.000014 could end up in a float that is supposed to equal 0
3. Is this IEEE standard well supported in your favorite high level language? Explain. Hint: Search “floating point errors” for ideas to check simple calculations. Turn in your code and results of execution.
   1. Floats are not supported to their fullest in my preferred programming language. If I correctly understand this article, I should be able to handle having non numerical values, which floats are unable to do in C++, Java (or any language that I know of).
   2. This being said, I am able to set values of all other values equal to zero or one, however, I am unable to set them to a pattern, for instance, 1, 0, 1, 0 without the use of a loop
   3. Furthermore, Mathematical operations are all supported by C++ in regards to floating point numbers.

//

// main.cpp

// floatTest

//

// Created by Jake Orben on 11/25/13.

// Copyright (c) 2013 Jake Orben. All rights reserved.

//

#include <iostream>

using namespace std;

int main(int argc, const char \* argv[])

{

float one;

float two;

float three;

float four;

float five;

int six;

double seven;

one = 1;

two = 123.13;

three = 12.3452;

four = 0.[12345274567](tel:12345274567" \t "_blank);

five = 3.345a;

// insert code here...

cout << "one " << one << ", two " << two << ", three " << three << ", four " << four << “, five ” << five;

one = one \* four;

three = two + one;

four = three;

cout << "one " << one << ", two " << two << ", three " << three << ", four " << four;

one = six \* four;

three = seven + one;

two = three \* one;

cout << "one " << one << ", two " << two << ", three " << three << ", four " << four;

return 0;

}

1. What should a computer scientist remember about floating point representation and support of the arithmetic operations? Consider a variety of professional settings, such as
   1. pharmaceutical industry
      * The key advantage in the pharmaceutical industry would be uniformity, using IEEE floating point would make a consistent dosage performance. The pharmaceutical industry needs to be especially careful with the use of floating point umbers. When creating dosages, the numbers need to be perfectly precise, otherwise, patients could be taking more or less of the medication that they need. Such an issue becomes relevant when dealing with micro-dosages, similar to micrograms, a small miscalculation could lead to a large problem. Furthermore, it should be noted that floating point numbers cannot be as precise as “doubles”, as they can only contain seven digits as a decimal. This could also result in imprecise measurements for medication.
   2. financial sector
      * Consistency is still the key advantage of using IEEE floating point to calculate values. This fact, however, is a double-edged sword, if the calculations are not precise there could be a cumulative miscalculation, leading to gross miscalculation in the stock exchange. The issues with the financial sector are similar to those with the pharmaceutical industry, the key issue be precision. As illustrated the movie “Office Space”, small amount of money could be stolen, using the fractions of percentages of pennies left over in a transaction. Furthermore, stock prices can be changed wildly based on only a handful of decimals, for instance when dealing with an index the DOW, the number of stocks in the system as well as the prices of the stocks could cause massive market fluctuation. Similar to the pharmaceutical industry, the limited, decimal availability of the floating point can cause issues when dealing with stock that have a precise value beyond what floating point numbers can represent.
   3. consumer product development
      * Similar to the previous two, the key issue with floating points is precision, if creating precision parts for a space shuttle or for a high performance car, precision is everything. In such cases, accuracy is measured in micro meters, tiny miscalculations on a system could lead to catastrophic failure. That being said, if thee calculations are correct, the IEEE standard will result in precision parts to be made consistently.

The above examples are not limiting; you may discuss the impact on any industry, but must use diverse examples.