Floating Point Computation - 2011/12 - Examples Sheets 1 and 2.

1. What is BCD (binary-coded decimal) ? Why do pocket calculators tend to work internally in BCD?

2. a) Perform round-to-even on the following decimal numbers 2.2, 3.5, 4.5, 5.6, 10.9. Answers should be natural numbers.

   b) Perform round-to-even applied to the following binary numbers 111.11, 111.101, 101.10, 110.1. Answers should be natural numbers expressed in binary.

   c) Apply round-to-even for 3 significant digits to the following 1.2345e5, 2.255e-10.

3. Give hand-crafted code that divides a floating point number by the constant 3. (Note: the slides give hand-crafted code that divides a 32-bit integer by the constant 10).

4. Given that most computer languages today support 32-bit signed integers and double-precision floating point, can it be argued that having the integers is silly since they are a subset of the doubles ?

5. Sketch a proof that integer comparison predicates can be applied to the bit patterns of IEEE unsigned floating point and mention any exceptions. (Or do a structured proof if feeling ambitious).

6. What do the following single precision IEEE bit patterns represent, where the msb of the first-listed byte is the sign bit ?

   a) 00 00 00 00

   b) 80 00 00 00

   c) BF 01 00 00

   d) 3F C0 00 00

   e) 04 04 04 00

7. What is $\log_{10}(2)$? How does this relate the number of bits in a binary integer to the number of digits in a decimal integer ? Give an example or two.

8. From the slides: 'For the example where `a, b & c` all have type float, give an example where `f(a*b+c)` generally gives a different answer to `{ float t=a*b; f(t+c); }`'.

9. Define machine epsilon and sketch a program to determine its value experimentally.

10. We should be well aware of the following basic, first-rate rules: relative errors add for mul/div and absolute errors sum for add/sub.

    a) What is a similar rule for the modulus operator ?

    b) Do these rules describe the average, worst-case or some other behaviour ?

    c) Try to describe, in the form of second-rate rules, what happens to relative errors when we add/sub and to absolute errors for mul/div ?

    Using decimal arithmetic and the rules you have just quoted or suggested, assuming each input value has been initially rounded to three significant figures, estimate the relative AND absolute errors when the following expressions are computed and finally stored using three significant figures. Illustrate an actual error assuming each input value was wrong by half a ulp ?

    d) (3.45 * 11.2) + 13.9

    e) (3.45 - 3.41) / 17.8

    Note: there are no nice second-rate rules, but it is of interest to see why not.

    Note: questions that are easier to answer with a calculator than without will not be set for Floating Point Tripos.

Floating Point Computation - 2011/12 - Examples Sheet 2 of 2.

1. Give four candidate control criteria that might be used to control the number of steps in an iteration and explain when they might be a good or bad choice ?

2. a) What is the formula for finding the square root of a number using Newton Raphson?

   b) When do we say that a numerical method is a second order method and when do we say an iteration has order of convergence 2 ?

   c) What is the order of convergence of Newton Raphson?

   d) Suppose the derivative of the target function is undesirably-expensive to compute: suggest a cheaper iteration based on Newton's method (makes the same graphical construction) and estimate/find/state its order of convergence.

3. Consider an iteration for the division n/d expressed using the following code fragment:

```
float n = 3223.231;
float d = 0.342;
for(...)
  {
    printf("%f  %f  %f\n", n, d, n/d);
    double f = 2.0 - d;
    n *= f;
    d *= f;
  }
```

   a) What happens? Does it work for a good range of n and d ?

   b) What is the order of convergence and why ?

   c) Is this a good method for division in general ?

4. Consider the matrix filled with Fibonacci numbers in the lecture notes:

$$\begin{pmatrix} 17711 & 10946 \\ 6765 & 4181 \end{pmatrix}$$

   they were inserted from the bottom right. What undesirable effect does this lead to when considered as a transformer on 2D co-ordinate spaces? Is it backwards stable? An array based on Fib(1,1) always has determinant 1, but what general propery of a 2×2 array leads to this undesirable effect?

5. Consider the quadratic $x^2 + 5x + 2 = (x + 0.438)(x + 4.561) = 0$.

   Two iterations can be considered (derive these):

$$\begin{aligned} x_{n+1} &= -2/(x_n + 5) \\ x_{n+1} &= -2/x_n - 5 \end{aligned}$$

   Is it the case that one finds one root and the other finds the other ? What happens if the starting guess for one is set at the solution of the other?

   [NB: You will not be expected to answer problems that are easier to solve using calculator/computer than by hand in Floating Point Tripos Examinations.]

```
Hint: here are the helpful runes for gnuplot:  gnuplot> plot -2/(x+5),x with lines
                                               gnuplot> plot -5-2/(x),x with lines
```

6. Consider the function arctan(y/x) giving the angle subtended at the origin for the point (x, y).

   a) The above function is 'two quadrant' only. Provide a more-complex implementation that involves '`if`' statement(s) and which gives a different answer in all four quadrants.

   b) Provide an approximate implementation of four-quadrant arctan that, instead of any sort of Taylor series, uses a few applications of the four basic arithmetic operators and also some '`if`' statements. You may return degrees instead of radians if you prefer.

   c) If a computer game needs to rapidly determine whether one point subtends a greater or smaller angle than another, is your implementation in part *b* suitable ?

7. Consider fixed-point decimal arithmetic:

   a) What are the following decimal numbers when rounded to even for a fixed-point decimal arithmetic system with eight places before the point and two after: 10.751, -100.755, -4032.382.

   b) Give an example, possibly such as an addition, in this number system where round to minus infinity gives a different answer from the normal round to even rule. Or say why not possible.

   c) Give an example of an interval arithmetic subtraction that illustrates the different rounding modes needed for the upper and lower bounds of the result.

   NB: An interval arithmetic system might equally well use floating point or fixed point for each member of a max/min pair.

8. a) What is meant by a chaotic system ? What feature of $x_{n+1} = 4 * x_n * (1 - x_n)$ leads to chaos ?

   ```
   NB: Verhulsts Logistic map: runes for gnuplot:  gnuplot> plot 4*x*(1-x), x  with lines
   ```

   b) What would we expect and like to see in terms of the numerical values of, and analytic expressions for, the partial derivatives in

   i) a well-behaved system,

   ii) a chaotic system, and

   iii) a system for suggesting re-balancing operations on an investment portfolio ?

Additional questions:

Also, make sure you understand the corner case described in (Java hangs when converting)

`http://www.exploringbinary.com/java-hangs-when-converting-2-2250738585072012e-308`

Also, compile the C or Java mini programs on the course web site and learn from them.