2007 Paper 10 Question 3

Floating-Point Computation

(*a*) A hypothetical (and practically rather useless) floating-point number representation inspired by the IEEE floating-point standards uses 6 bits— one bit for sign, three bits for exponent and two (stored) bits for mantissa (significand). Assuming that 1.0 is represented in this format as 0:011:00 give the values, in decimal notation (fractions are acceptable), of all the other non-negative floating-point values in this representation. You do not need to give details of denormalised numbers or NaNs.   [10 marks]

(*b*) Explain the following terms:

   (*i*)   absolute error;

   (*ii*) relative error;

   (*iii*) rounding error;

   (*iv*) truncation error;

   (*v*)   ill-conditionedness.   [5 marks]

(*c*) Assuming the floating-point representation for type `float` has b bits in its mantissa (significand), what can be said about the output of the following program?

```
float f = 10.0/3.0;
for (i=0; i<100; i++)
{ int v = (int)f;        /* get integer part of f */
  printf("%d\n", v);     /* print it */
  f = (f - v) * 10;
}
```

Discuss how accurately `f` represents 10.0/3.0 at the start of each iteration and explain which operation(s) represent the main loss of accuracy in `f` on each iteration. (You may assume that 10 significant bits of accuracy is approximately 3 decimal digits of accuracy.)   [5 marks]

1