Floating-Point Computation

(a) Write a function in a programming language of your choice that takes a (32-bit IEEE format) float and returns a float with the property that: given zero, infinity or a positive normalised floating-point number then its result is the smallest normalised floating-point number (or infinity if this is not possible) greater than its argument. You may assume functions f2irep and irep2f which map between a float and the same bit pattern held in a 32-bit integer. [6 marks]

(b) Briefly explain how this routine can be extended also to deal with negative floating-point values, remembering that the result should always be greater than the argument. [2 marks]

(c) Define the notions of rounding error and truncation error of a floating-point computation involving a parameter $h$ that mathematically should tend to zero. [2 marks]

(d) Given a function $f$ implementing a differentiable function that takes a floating-point argument and gives a floating-point result, a programmer implements a function

$$f'(x) \approx \frac{f(x + h) - f(x - h)}{2h}$$

to compute its derivative. Using a Taylor expansion or otherwise, estimate how rounding and truncation errors depend on $h$. You may assume that all mathematical derivatives of $f$ are within an order of magnitude of 1.0. [8 marks]

(e) Suggest a good value for $h$ given a double-precision floating-point format that represents approximately 15 significant decimal figures. [2 marks]