1. [5 points]

    unsigned compare(int x, int y)
    {
        return !((x^y));
    }

2. [5 points]

    void set_bits(unsigned x, unsigned l, unsigned r)
    {
        int i;
        for (i = l; i <= r; i--)
            x |= (1<<i);
    }

3. [2 points]

    The function counts the number of bits set to 1 in k;

4. [3 points]

    N-bit number means it can present a range 0 → 2^{N-1}
    Let’s assume x = 2^{N-1}
    Now, the largest number that a*b+c can produce is: x^2+x
    After doing some simplification, we reach: 2^N(2^{N-1})
    This is the largest number we can get. To know the number of bits, we need to take log to the base 2 of that number: log[2^N(2^{N-1})]
    = log[2^N] + log[(2^{N-1})] ~ 2N

5. [9 points] From what we studied in class, the bias = 2^{w-1}-1 where w is the number of bits in the bias. In this problem w = 3, so the bias = 3;
   a) The smallest non-zero will come from a denormalized form, because this form is made to represent 0 and very small numbers.
      In binary, this number will be 0 000 1
      sign = +ve
      E = 1 – bias = 1 – 3 = -2
      Mantissa = 0.1 = 0.5
      The number in decimal is then: + 2^{-2} * 0.5 = +2^{-3}
b) Largest is: 0 110 1 (note that the exponent cannot be 111 otherwise we will be in the special values part).
   Sign is +ve
   Exponent is 6-3 = 3
   Mantissa = 1.5
   So the number is   \(+2^3\) (1.5)

6. [6 points]

   a) \(y = 7*x;\)

       \(y = (x<<3) - x;\)

   b) \(y = 27*x;\)

       \(y = (x<<5) - (x<<2) - x;\)

   c) \(y = 67*x;\)

       \(y = (x<<6) + (x<<1) + x;\)