1. [5 points]
   ```c
   unsigned compare(int x, int y){
      return !(x^y);
   }
   ```

2. [5 points]
   ```c
   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 \( +2^3 \times (1.5) \)

6. [6 points]

   a) \( y = 7 \times x; \)
      \[ y = (x << 3) - x; \]

   b) \( y = 27 \times x; \)
      \[ y = (x << 5) - (x << 2) - x; \]

   c) \( y = 67 \times x; \)
      \[ y = (x << 6) + (x << 1) + x; \]
7. [5 points (-0.5 for each mistake)]

<table>
<thead>
<tr>
<th>Expression</th>
<th>Decimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>zero</td>
<td>0</td>
<td>000 000</td>
</tr>
<tr>
<td>-</td>
<td>-6</td>
<td>111 010</td>
</tr>
<tr>
<td>-</td>
<td>+18</td>
<td>01 0010</td>
</tr>
<tr>
<td>ux</td>
<td>47</td>
<td>101 111</td>
</tr>
<tr>
<td>y</td>
<td>-3</td>
<td>111 101</td>
</tr>
<tr>
<td>Tmax</td>
<td>+31</td>
<td>011 111</td>
</tr>
<tr>
<td>-TMin</td>
<td>-32</td>
<td>100 000</td>
</tr>
</tbody>
</table>

-TMin and Tmax are the minimum and maximum two’s complement numbers.

For signed two’s complement the range goes from $-2^{n-1} \rightarrow +2^{n-1}-1$

For unsigned the range goes from $0 \rightarrow +2^{n-1}$

In this problem $n = 6$

8. [A: 1 point, B: 2 points, C: 2 points]

A. On a x86 32-bit machine, Alice intends to use the expression

\[ \text{if} \ (x \ \& \ \text{mask}) \neq 0 \]  

to test if the 5th bit of $x$ from the right is one or not. (The rightmost bit of $x$ is considered as the 0-th bit). The value of mask should be \[ \boxed{32} \] (decimal).

B. Which of the following expressions generate the desired mask value in Question (A)? Select all that apply.

(a) $1 \ll 6$
(b) $1 \ll 5$
(c) $\sim(1\ll 6)$
(d) $\sim(1\ll 5)$
(e) $1 \gg 26$
(f) $1 \gg 27$

C. Please give the expression which sets the 5th bit of $x$ to be one and leave the rest of the bits of $x$ unchanged. Your expression should only use the mask value in Question (A) and no other constants.

\[ \boxed{32 | x} \]