Exercise – Binary

1. Convert the following from decimal to binary
   * 1
   * 42
   * 256
   * 4,294,967,296
2. Convert the following from binary to decimal
   * 10000000
   * 10101010
   * 11110000
   * 11001100
3. Solve these binary arithmetic problems
   * 111 + 111
   * 1010 + 1010
   * 11101 + 1010
   * 1101 - 11
   * 10001 - 100
   * 101 x 10
   * 1011 x 11
   * 1101 / 11
4. In many places, numbers stored in computers are displayed as hexadecimal (base 16). What advantages do hex numbers have over decimal and binary numbers respectively when displaying numbers stored in a computer?
5. Using Two's complement convert the following between decimal and binary (for binary use 1 signed byte)

* 10000000
* 10101010
* 11110000
* 11001100
* -16
* 128
* -128
* -123

1. What do each of these evaluate to?
   * 11111 | 11111
   * 11111 ^ 11111
   * 10101 & 11111
   * 10101 | 11111
   * 00000 ^ 11111
   * 1 << 3
   * 100 >> 2
   * ~10101
   * 100 << 1
   * 1010 >> 1
   * ~11111
2. A true/false value can be stored in a single bit – zero for false, one for true. However, the Boolean type in C is a full byte big – 8 bits. This means it is possible to store 8 bits in a single byte. Using bitwise operators, how might you
   * Set an single bit to 0
   * Set an single bit to 1
   * Check the value of a single bit
3. Implement the following functions in C++
   * bool IsLeftMostBitSet(unsigned int value)
     + Returns true if the left most (the most significant) bit of value is set and false otherwise
   * bool IsRightMostBitSet(unsigned int value)
     + Returns true if the right most (the least significant) bit of value is set and false otherwise
   * bool IsBitSet(unsigned int value, unsigned char bit\_to\_check)
     + Returns true if the asked for bit is set, and false otherwise. bit\_to\_check is zero indexed from the right most bit. i.e 0 is the right most bit and 31 is the left most.
   * int GetRightMostSetBit(unsigned int value)
     + This function returns the index of the right most bit set to 1 in value
     + If no bits are set, it returns -1
     + For example
       - 00000001 would return 0
       - 10011100 would return 2
       - 01010000 would return 4
       - 00000000 would return -1
   * void PrintBinary(unsigned char value)
     + Prints value to the console as a binary number
   * CHALLENGE: bool IsPowerOf2(unsigned int value)
     + Returns true of value is a power of 2 and false otherwise. Use only bitwise and arithmetic operators.