Integers in Java are stored using two's-compliment representation. The bits have place value 1, 2, 4, 8, 16, 32, …, from right-to-left, except the leftmost bit is worth a negative value. For example, in 8-bit two's-compliment 10000011 represents -128 + 2 + 1 = -125.

1) What are the values of the following 8-bit strings when interpreted using two's compliment? Write each as a sum of powers of two and as a total, like the example.

   00001111 = 8+4+2+1 = 15

   10101010

   01010101

2) Write each of the following using 8-bit two's compliment. (If you are not sure how to do this see the note at the end of this worksheet.)

   5 = 4+1 = 00000101

   55

   127

   -55

   -127

3) In Java, variables of the following types represent integers. For each, how many bits are there in variables of the type, what is the minimum value, and what is the maximum value. If you do not know, you may need to search the internet or a textbook.

   `short`

   `int`

   `long`

   `byte`

Recall that a *literal* is a text representation of a value and is used to assign a particular value to a variable. There are three ways of writing an integer literal.

```
int normal = 123;         // Most common, decimal
int hex = 0x123ABC;       // Less common, hexadecimal
int binary = 0b01010101; // Less common, binary
```

The hexadecimal (often just called hex) and binary forms are typically only used when it is important what bit pattern is being placed in a variable.

To convert a number into its hex form, write it out in binary. If the number of binary digits is not a multiple of four, add extra zeros to the beginning. Now intepret each group of four bits into a single hexadecimal digit. For example, 11000011 is broken into 1100 0011, which is C 3 in hexadecimal. For the numbers 10, 11, 12, 13, 14, 15, hexadecimal uses A, B, C, D, E, F instead.

4) Write each of the following as both hexadecimal and binary integer literals. Don't forget the 0x and 0b which tells Java which you are using.

   5

   55

   127

   -55

   -127

NOTE: Here is how I convert decimal to binary. There are other ways, but this is my favorite.

Decimal to Binary(dec):
  if dec < 0
    put a 1 in the leftmost position of binary
    subtract its (negative) value from dec, making dec positive
  while dec ≠ 0
    find x, the largest power of two not bigger than dec
    put a 1 in the binary place-value position representing x
    subtract x from dec
  all remaining positions in the binary get 0