Questions

Question 1

Counting practice: count from zero to thirty-one in binary, octal, and hexadecimal:

	Binary	Octal	Hex
Zero			
One			
Two			
Three			
Four			
Five			
Six			
Seven			
Eight			
Nine			
Ten			
Eleven			
Twelve			
Thirteen			
Fourteen			
Fifteen			

	Binary	Octal	Hex
Sixteen			
Seventeen			
Eighteen			
Nineteen			
Twenty			
Twenty one			
Twenty two			
Twenty three			
Twenty four			
Twenty five			
Twenty six			
Twenty seven			
Twenty eight			
Twenty nine			
Thirty			
Thirty one			

Question 2

Add the following binary numbers:

```
  10010        1011101        10011
+ 1100       + 1000000      + 1111101

  10011001     11000011       1001100
+   100111   +   101111     + 1100101
```

Question 3

If the numbers sixteen and nine are added in binary form, will the answer be any different than if the same quantities are added in decimal form? Explain.

Question 4

What is the *one's complement* of a binary number? If you had to describe this principle to someone who just learned what binary numbers are, what would you say?

Determine the one's complement for the following binary numbers:

- 10001010_2
- 11010111_2
- 11110011_2
- 11111111_2
- 11111_2
- 00000000_2
- 00000_2

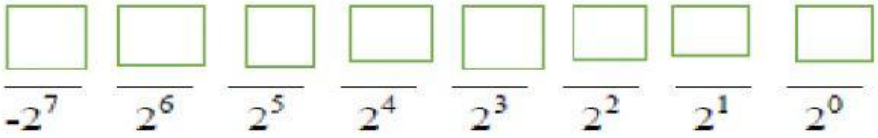
Question 5

Determine the *two's complement* of the binary number 01100101_2. Explain how you did the conversion, step by step.

Next, determine the two's complement representation of the quantity *five* for a digital system where all numbers are represented by four bits, and also for a digital system where all numbers are represented by eight bits (one *byte*). Identify the difference that "word length" (the number of bits allocated to represent quantities in a particular digital system) makes in determining the two's complement of any number.

Question 6

In a computer system that represents all integer quantities using two's complement form, the most significant bit has a negative place-weight. For an eight-bit system, the place weights are as follows:



Given this place-weighting, convert the following eight-bit two's complement binary numbers into decimal form:

- 01000101_2 =
- 01110000_2 =
- 11000001_2 =
- 10010111_2 =
- 01010101_2 =
- 10101010_2 =
- 01100101_2 =

Question 7

In an eight-bit digital system, where all numbers are represented in two's complement form, what is the largest (most positive) quantity that may be represented with those eight bits? What is the smallest (most negative) quantity that may be represented? Express your answers in both binary (two's complement) and decimal form.

Question 8

Two's complement notation really shows its value in binary addition, where positive and negative quantities may be handled with equal ease. Add the following byte-long (8 bit) two's complement numbers together, and then convert all binary quantities into decimal form to verify the accuracy of the addition:

```
 00110101      01110110      00111101
+00001100     +00000010     +11111011
```

```
 00001010      11111110      11111110
+10010101     +11011101     +11111101
```

Question 9

Add the following eight-bit two's complement numbers together, and then convert all binary quantities into decimal form to verify the accuracy of the addition:

```
 10110111      00111101      11111011
+01110110     +00111011     +11111011
```

```
 10000001      01111011      01111111
+10010001     +00111101     +10000001
```

Question 10

How is it possible to tell that *overflow* has occurred in the addition of binary numbers, without converting the binary sums to decimal form and having a human being verify the answers?

Question 11

What is a *floating-point* number in a digital system?