1.

1 = 1 = (1)

42 = 101010 = (32 + 8 + 2)

256 = 100000000 = (256)

4294967296 = 100000000000000000000000000000000 = (4294967296)

2.

10000000 = (128) = 128

10101010 = (128 + 32 + 8 + 2) = 170

11110000 = (128 + 64 + 32 + 16) = 240

11001100 = (128 + 64 + 8 + 4) = 204

3.

111 + 111 = (4 + 2 + 1) + (4 + 2 + 1) = 7 + 7 = 14 = (8 + 4 + 2) = 1110

1010 + 1010 = (8 + 2) + (8 + 2) = 10 + 10 = 20 = (16 + 4) = 10100

11101 + 1010 = (16 + 8 + 4 + 1) + (8 + 2) = 29 + 10 = 39 = (32 + 4 + 2 + 1) = 100111

1101 - 11 = (8 + 4 + 1) - (2 + 1) = 13 - 3 = 10 = (8 + 2) = 1010

10001 - 100 = (16 + 1) - (4) = 17 - 4 = 13 = (8 + 4 + 1) = 1101

101 \* 10 = (4 + 1) \* (2) = 5 \* 2 = 10 = (8 + 2) = 1010

1011 \* 11 = (8 + 2 + 1) \* (2 + 1) = 11 \* 3 = 33 = (32 + 1) = 100001

1101 / 11 = (8 + 4 + 1) / (2 + 1) = 13 / 3 = 4 = 100

4.

A: Larger numbers can be displayed using the same number of digits.

B: Hexadecimal is easier to convert to and from binary than decimal because it is a power of 2.

5.

10000000

invert = 01111111 = (1 + 2 + 4 + 8 + 16 + 32 + 64) = 127

add 1 = 10000000 = 128

10000000 = -128

10101010

invert = 01010101 = (1 + 4 + 16 + 64) = 85

add 1 = 86

10101010 = -86

11110000

invert = 00001111 = (1 + 2 + 4 + 8) = 15

add 1 = 16

11110000 = -16

11001100

invert = 00110011 = (1 + 2 + 16 + 32) = 51

add 1 = 52

11001100 = -52

-16

16 = 00010000

invert = 11101111

add 1 = 11110000

-16 = 11110000

128

-128 = 10000000

invert = 01111111

add 1 = 10000000

128 = 10000000

-128

128 = 10000000

invert = 01111111

add 1 = 10000000

-128 = 10000000

-123

123 = 01111011

invert = 10000100

add 1 = 10000101

-123 = 10000101

6.

11111 | 11111 = 11111

11111 ^ 11111 = 00000

10101 & 11111 = 10101

10101 | 11111 = 11111

00000 ^ 11111 = 11111

1 << 3 = 1000

100 >> 2 = 1

~10101 = 01010

100 << 1 = 1000

1010 >> 1 = 0101

~11111 = 00000

7.

To set a single bit to to 0 you could use the AND operator along with a byte that has a 0 in the bit you want to change and 1s in the rest.

10010101 & 11101111 = 10000101 (sets the 4th bit to 0)

To set a single bit to 1 you could do use the OR operator with a byte that has a 1 in the bit you want to change and 0s in the rest.

10010101 | 01000000 = 11010101 (sets the 2nd bit to 1)

To check the value of a specific bit you could use the AND operator with a byte that has a 1 in the bit that you want to check and 0s in the rest.

The resulting byte can be used as a bool because any number other than 0 counts as true.

10010101 & 00001000 = 00000000 = false (the 5th bit is not set)

8.

I included these functions in my maths library.