F.2 Chapter 2 Solutions

2.1 The answer is 2^n

2.2 For 26 characters, we need at least 5 bits. For 52 characters, we need at least 6 bits.

2.3 (a) For 400 students, we need at least 9 bits.

(b) $2^9 = 512$, so 112 more students could enter.

2.4 2^n integers can be represented. The range would be 0 to (2^n) - 1.

2.5 If each number is represented with 5 bits,

```
 7 = 00111 in all three systems
-7 = 11000 (1's complement)
   = 10111 (signed magnitude)
   = 11001 (2's complement)
```

2.6 100000.

2.7 Refer the following table:

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

2.8 The answers are:

(a) 127 in decimal, 01111111 in binary.

(b) -128 in decimal, 10000000 in binary.

(c) (2^{n-1})-1

(d) -(2^{n-1})

2.9 Avogadro's number (6.02×10^{23}) requires 80 bits to be represented in two's complement binary representation.

2.10 The answers are:

(a) -6
(b) 90
(c) -2
(d) 14803

2.11 (a) 01100110
(b) 01000000
(c) 00100001
(d) 10000000
(e) 01111111

2.12 It is a multiple of 4.

2.13 (a) 11111010
(b) 00011001
(c) 11111000
(d) 00000001

2.14 (a) 1100
(b) 1010
(c) 1111
(d) 01011
(e) 10000

2.15 Dividing the number by two.

2.16 (a) 11111111 (binary) or -0 (decimal)
(b) 10001110 (binary) or -14(decimal)
(c) 00000000 (binary) or 0 (decimal)

2.17 (a) 1100 (binary) or -4 (decimal)
(b) 01010100 (binary) or 84 (decimal)
(c) 0011 (binary) or 3 (decimal)
(d) 11 (binary) or -1 (decimal)

2.18 The answers are:

(a) 1100 (binary) or 12 (decimal)

(b) 1011000 (binary) or 88 (decimal)
(c) 1011 (binary) or 11 (decimal)
(d) 11 (binary) or 3 (decimal)

2.19 11100101, 1111111111100101, 11111111111111111111111111100101. Sign extension does not affect the value represented.

2.20 (a) 1100 + 0011 = 1111
-4 + 3 = -1
(b) 1100 + 0100 = 0000
-4 + 4 = 0
(c) 0111 + 0001 = 1000 OVERFLOW!
7 + 1 = -8
(d) 1000 - 0001 = 1000 + 1111 = 0111 OVERFLOW!
-8 - 1 = -8 + (-1) = 7
(e) 0111 + 1001 = 0000
7 + -7 = 0

2.21 Overflow has occurred if both operands are positive and the result is negative, or if both operands are negative and the result is positive.

2.22 Any two 16-bit 2's complement numbers that add to more than +32767 or less than -32768 would be correct.

2.23 Overflow has occurred in an unsigned addition when you get a carry out of the leftmost bits.

2.24 Any two 16-bit unsigned numbers that add to more than 65535 would be correct.

2.25 Because their sum will be a number which if positive, will have a lower magnitude (less positive) than the original postive number (because a negative number is being added to it), and vice versa.

2.26 (a) 7 bits.
(b) 63 in decimal (0111111 in binary)
(c) 127 in decimal (1111111 in binary)

2.27 The problem here is that overflow has occurred as adding 2 positive numbers has resulted in a negative number.

2.28 When all of the inputs are 1.

2.29 Refer to the following table:

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

2.30 (a) 01010111
(b) 100
(c) 10100000
(d) 00010100
(e) 0000
(f) 0000

2.31 When atleast one of the inputs is 1.

2.32 Refer to the following table:

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

2.33 (a) 11010111
(b) 111
(c) 11110100
(d) 10111111
(e) 1101
(f) 1101

2.34 (a) 0111
(b) 0111
(c) 1101
(d) 0110

2.35 The masks are used to set bits (by ORing a 1) and to clear bits (by ANDing a 0).

2.36 (a) AND with 11111011
(b) OR with 01000100
(c) AND with 00000000
(d) OR with 11111111
(e) AND the BUSYNESS pattern with 00000100 to isolate b2. Then add that result to itself 5 times.

2.37 [(n AND m AND (NOT s)) OR ((NOT n) AND (NOT m) AND s)] AND 1000

2.38 Let N = n & 1000, M = m & 1000
Overflow = (N AND M) OR ((N OR M) AND ((NOT s) AND 1000))

2.39 (a) 0 10000000 11100000000000000000000
(b) 1 10000100 10111010111000000000000
(c) 0 10000000 10010010000111111011011
(d) 0 10001110 11110100000000000000000

2.40 (a) 2
(b) -17
(c) Positive infinity. NOTE: This was not explained in the text.
(d) -3.125

2.41 (a) 127
(b) -126

2.42 The ASCII values are being added, rather than the integer values. (ASCII ”5” is 53 in decimal, and ASCII ”8” is 56 in decimal, adding to 109, which is ASCII ”m”.)

2.43 (a) Hello!
(b) hELLO!
(c) Computers!
(d) LC-2

2.44 Add 0011 0000 (binary) or x30.

2.45 (a) xD1AF
(b) x1F
(c) x1
(d) xEDB2

2.46 (a) 0001 0000
(b) 1000 0000 0001
(c) 1111 0111 0011 0001
(d) 0000 1111 0001 1110 0010 1101
(e) 1011 1100 1010 1101

2.47 (a) -16
(b) 2047
(c) 22
(d) -32768

2.48 (a) x100
(b) x6F
(c) x75BCD15

(d) xD4

2.49 (a) x2939

(b) x6E36

(c) x46F4

(d) xF1A8

(e) The results must be wrong. In (3), the sum of two negative numbers produced a positive result. In (4), the sum of two positive numbers produced a negative result. We call such additions OVERFLOW.

2.50 (a) x5468

(b) xBBFD

(c) xFFFF

(d) x32A3

2.51 (a) x644B

(b) x4428E800

(c) x48656C6C6F

2.52 Refer to the table below.

	x434F4D50	x55544552
Unsigned Binary	1,129,270,608	1,431,586,130
1's Complement	1,129,270,608	1,431,586,130
2's Complement	1,129,270,608	1,431,586,130
IEEE 754 floating point	207.302001953125	14,587,137,097,728
ASCII String	COMP	UTER

2.53 Refer to the table below:

A	B	Q1	Q2
0	0	1	0
0	1	1	1
1	0	1	1
1	1	0	1

Q2 = A OR B

2.54 Refer to the table below:

X	Y	Z	Q1	Q2
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

2.55 (a) 63

(b) 4^n - 1

(c) 310

(d) 222

(e) 11011.11

(f) 0100 0001 1101 1110 0000 0000 0000 0000

(g) 4^{4m}

2.56 - 1.101 x $2^{(12-7)}$ = -52