Bits and Bytes

1 Computer Memory

All data is represented inside the computer as sequences of bits. A *bit* is the basic unit of digital information and can assume one of two values, "off" or "on". A *byte* is a sequence of 8 bits. Computer memory consists of a very long sequence of bytes. For example, a gigabyte memory chip is a physical device that contains $2^{30} = 1{,}073{,}741{,}824$ bytes, or 8,589,934,592 bits.

In C, sequences of bytes are used to represent integers, real numbers, characters, and so forth. We normally do not distinguish between an object and its underlying representation, so we say that the `int` variable `x` contains the number 123 as if the computer could really store numbers (and not just bits). However, sometimes we want to be more precise and to talk about the representing bits directly. For that, we need unambiguous notation and terminology.

2 Describing Bit Sequences

For single bits, we use the characters '0' and '1' to describe the values "off" and "on", respectively. To describe a sequence of bits, we can use a string of 0's and 1's. However, long strings of 0's and 1's are not very easy for humans to deal with, so we will use other notations as well.

We can pretend a bit string is a binary (base 2) number. We can then uniquely describe it by giving its length and value. For example, the bit string 010010 is the unique string of length 6 and integer value 18, since $(010010)_2 = 18$. Note that it's not sufficient to just give the value since one cannot tell from that how many leading 0's the bit string has. For example, 0000010010 is a different bit string from 010010, but both have value 18. Because of this identification between bit strings and numbers, people often say that everything inside a computer is represented by numbers, but this is really a misnomer.

The problem with describing a bit string by a length and value is that it is not very easy to figure out which bits are on or off just by knowing a value expressed as a decimal numeral. If I ask you the question, "How many 1's are there in the bit string with value 1234?", you'd have to do quite a bit of work to answer it. Of course, if you express the value as a binary numeral, then the number *is* the bit string (without leading 0's), but now we've come full circle and again have an unwieldy representation. It turns out that expressing the value in base 16 gives a nice compromise between succinctness and ease of decoding the underlying bits.

3 Hex numbers

A hex number is a number represented in base 16. A hex digit is a number between 0 and 15, just as a decimal digit is a number between 0 and 9. We can use the symbols '0' through '9' as usual to represent the first 10 hex digits, but we need additional symbols for $10, \ldots, 15$. By convention, we use the first six letters of the alphabet for that purpose, so `a` $= 10$, `b` $= 11$, `c` $= 12$, `d` $= 13$, `e` $= 14$, and `f` $= 15$. Upper case letters '`A`', ..., '`F`' are often used instead, and C allows both.

Numbers bigger than 15 are represented in hex by a string of hex digits, just as decimal numbers bigger than 9 are represented by a string of decimal digits. The place values of a hex number are powers of 16. Thus, $(\texttt{3a})_{16} = 3 * 16 + 10 = 58$ and $(\texttt{abc})_{16} = 16^2 * 10 + 16 * 11 + 12 = 2748$.

Returning to the previous example, the 6-bit sequence whose value is 18 can be described as the 6-bit sequence whose value in hex is 12. To distinguish hex numerals from decimal numerals, we follow the C convention and prefix them with "0x". Thus, we can say that 0x12 = 18. The nice thing about hex is that each hex digit corresponds to 4 bits of the underlying bit string, so to answer the question of what bit string is represented by 0x12, one simply replaces each of the digits by the 4-bit binary number of the same value. Since 0x1 = $(0001)_2$ and 0x2 = $(0010)_2$, it follows that 0x12 represents the bit string 00010010. Since we're also told the length is 6, we know the top two zeros should be discarded, giving 010010.

The process of going from a bit string to its hex representation is also simple. First divide the bit string into groups of 4 bits each, starting from the right. Then replace each group by its corresponding hex digit as given in Table 1. Equivalently, find the value of the 4-bit group as a binary number and replace it by the hex digit of same value.

Table 1: Representing bit strings by hex digits

Bit String	Hex Digit	Bit String	Hex Digit
0000	0	1000	8
0001	1	1001	9
0010	2	1010	a
0011	3	1011	b
0100	4	1100	c
0101	5	1101	d
0110	6	1110	e
0111	7	1111	f