节。在小端法机器上，它将按照从最低有效字节到最高有效字节的顺序列出字节。在大端法机器上，它将按照从最高有效字节到最低有效字节的顺序列出字节。

2.6 这又是一个练习从十六进制到二进制转换的机会。同时也让你思考整数和浮点表示。我们将在本章后面更加详细地研究这些表示。

A. 利用书中示例的符号，我们将两个串写成：

```
   0   0   3   5   9   1   4   1
00000000001101011001000101000001
          ********************
   4   A   5   6   4   5   0   4
01001010010101100100010100000100
```

B. 将第二个字相对于第一个字向右移动 2 位，我们发现一个有 21 个匹配位的序列。

C. 我们发现除了最高有效位 1，整数的所有位都嵌在浮点数中。这正好也是书中示例的情况。另外，浮点数有一些非零的高位不与整数中的高位相匹配。

2.7 它打印 61　62　63　64　65　66。回想一下，库函数 strlen 不计算终止的空字符，所以 show_bytes 只打印到字符 '\f'。

2.8 这是一个帮助你更加熟悉布尔运算的练习。

运算	结果	运算	结果
a	[01101001]	*a*&*b*	[01000001]
b	[01010101]	*a* | *b*	[01111101]
~*a*	[10010110]	*a* ^ *b*	[00111100]
~*b*	[10101010]		

2.9 这个问题说明了怎样用布尔代数来描述和解释现实世界的系统。我们能够看到这个颜色代数和长度为 3 的位向量上的布尔代数是一样的。

A. 颜色的取补是通过对 *R*、*G* 和 *B* 的值取补得到的。由此，我们可以看出，白色是黑色的补，黄色是蓝色的补，红紫色是绿色的补，蓝绿色是红色的补。

B. 我们基于颜色的位向量表示来进行布尔运算。据此，我们得到以下结果：

蓝色(001)　|　绿色(010)　=　蓝绿色(011)

黄色(110)　&　蓝绿色(011)=　绿色(010)

红色(100)　^　紫红色(101)=　蓝色(001)

2.10 这个程序依赖于两个事实，EXCLUSIVE-OR 是可交换的和可结合，以及对于任意的 *a*，有 *a*^*a*=0。

步骤	**x*	**y*
初始	*a*	*b*
步骤1	*a*	*a*^*b*
步骤2	*a* ^ (*a* ^ *b*) = (*a* ^ *a*) ^ *b* =*b*	*a*^*b*
步骤3	*b*	*b*^ (*a* ^ *b*) = (*b* ^ *b*) ^ *a* = *a*

某种情况下这个函数会失败，参见练习题 2.11。

2.11 这个题目说明了我们的原地交换例程微妙而有趣的特性。

A. first 和 last 的值都为 *k*，所以我们试图交换正中间的元素和它自己。

B. 在这种情况中，inplace_swap 的参数 x 和 y 都指向同一个位置。当计算 *x^*y 的时候，我们得到 0。然后将 0 作为数组正中间的元素，而后面的步骤一直都把这个元素设置为 0。我们可以看到，练习题 2.10 的推理隐含地假设 x 和 y 代表不同的位置。

C. 将 reverse_array 的第 4 行的测试简单地替换成 first<last，因为没有必要交换正中间的元素和它自己。

2.12 这些表达式如下：