当执行C程序时，不会将溢出作为错误而发信号。不过有的时候，我们可能希望判定是否发生了溢出。

原理：检测无符号数加法中的溢出

对在范围 $0 \leqslant x, y \leqslant UMax_w$ 中的 x 和 y，令 $s \doteq x +^{\mathrm{u}}_w y$。则对计算 s，当且仅当 $s < x$（或者等价地 $s < y$）时，发生了溢出。

作为说明，在前面的示例中，我们看到 $9 +^{\mathrm{u}}_4 12 = 5$。由于 $5 < 9$，我们可以看出发生了溢出。

推导：检测无符号数加法中的溢出

通过观察发现 $x + y \geqslant x$，因此如果 s 没有溢出，我们能够肯定 $s \geqslant x$。另一方面，如果 s 确实溢出了，我们就有 $s = x + y - 2^w$。假设 $y < 2^w$，我们就有 $y - 2^w < 0$，因此 $s = x + (y - 2^w) < x$。 ■

练习题 2.27 写出一个具有如下原型的函数：

```
/* Determine whether arguments can be added without overflow */
int uadd_ok(unsigned x, unsigned y);
```

如果参数 x 和 y 相加不会产生溢出，这个函数就返回1。

模数加法形成了一种数学结构，称为阿贝尔群（Abelian group），这是以丹麦数学家 Niels Henrik Abel（1802～1829）的名字命名。也就说，它是可交换的（这就是为什么叫"abelian"的地方）和可结合的。它有一个单位元0，并且每个元素有一个加法逆元。让我们考虑 w 位的无符号数的集合，执行加法运算 $+^{\mathrm{u}}_w$。对于每个值 x，必然有某个值 $-^{\mathrm{u}}_w x$ 满足 $-^{\mathrm{u}}_w x +^{\mathrm{u}}_w x = 0$。该加法的逆操作可以表述如下：

原理：无符号数求反

对满足 $0 \leqslant x < 2^w$ 的任意 x，其 w 位的无符号逆元 $-^{\mathrm{u}}_w x$ 由下式给出：

$$-^{\mathrm{u}}_w x = \begin{cases} x, & x = 0 \\ 2^w - x, & x > 0 \end{cases} \tag{2.12}$$

该结果可以很容易地通过案例分析推导出来：

推导：无符号数求反

当 $x=0$ 时，加法逆元显然是0。对于 $x>0$，考虑值 $2^w - x$。我们观察到这个数字在 $0 < 2^w - x < 2^w$ 范围之内，并且 $(x + 2^w - x) \bmod 2^w = 2^w \bmod 2^w = 0$。因此，它就是 x 在 $+^{\mathrm{u}}_w$ 下的逆元。 ■

练习题 2.28 我们能用一个十六进制数字来表示长度 $w=4$ 的位模式。对于这些数字的无符号解释，使用等式（2.12）填写下表，给出所示数字的无符号加法逆元的位表示（用十六进制形式）。

x		$-^{\mathrm{u}}_4 x$	
十六进制	十进制	十进制	十六进制
0			
5			
8			
D			
F			

2.3.2 补码加法

对于补码加法，我们必须确定当结果太大（为正）或者太小（为负）时，应该做些什么。