优先级	运算符	名称或含义	使用形式	结合方向	说明
1	[]	数组下标	数组名[常量表达式]	左到右	
	()	圆括号	(表达式)/函数名(形参表)		
	.	成员选择（对象）	对象.成员名		
	->	成员选择（指针）	对象指针->成员名		
2	-	负号运算符	-表达式	右到左	单目运算符
	(类型)	强制类型转换	(数据类型)表达式		
	++	自增运算符	++变量名/变量名++		单目运算符
	--	自减运算符	--变量名/变量名--		单目运算符
	*	取值运算符	*指针变量		单目运算符
	&	取地址运算符	&变量名		单目运算符
	!	逻辑非运算符	!表达式		单目运算符
	~	按位取反运算符	~表达式		单目运算符
	sizeof	长度运算符	sizeof(表达式)		
3	/	除	表达式/表达式	左到右	双目运算符
	*	乘	表达式*表达式		双目运算符
	%	余数（取模）	整型表达式/整型表达式		双目运算符
4	+	加	表达式+表达式	左到右	双目运算符
	-	减	表达式-表达式		双目运算符
5	<<	左移	变量<<表达式	左到右	双目运算符
	>>	右移	变量>>表达式		双目运算符
6	>	大于	表达式>表达式	左到右	双目运算符
	>=	大于等于	表达式>=表达式		双目运算符
	<	小于	表达式<表达式		双目运算符
	<=	小于等于	表达式<=表达式		双目运算符
7	==	等于	表达式==表达式	左到右	双目运算符
	!=	不等于	表达式!= 表达式		双目运算符
8	&	按位与	表达式&表达式	左到右	双目运算符
9	^	按位异或	表达式^表达式	左到右	双目运算符
10	|	按位或	表达式|表达式	左到右	双目运算符
11	&&	逻辑与	表达式&&表达式	左到右	双目运算符
12	||	逻辑或	表达式||表达式	左到右	双目运算符
13	?:	条件运算符	表达式 1? 表达式 2: 表达式 3	右到左	三目运算符
14	=	赋值运算符	变量=表达式	右到左	
	/=	除后赋值	变量/=表达式		
	*=	乘后赋值	变量*=表达式		
	%=	取模后赋值	变量%=表达式		
	+=	加后赋值	变量+=表达式		
	-=	减后赋值	变量-=表达式		
	<<=	左移后赋值	变量<<=表达式		
	>>=	右移后赋值	变量>>=表达式		

	&=	按位与后赋值	变量&=表达式		
	^=	按位异或后赋值	变量^=表达式		
	|=	按位或后赋值	变量|=表达式		
15	,	逗号运算符	表达式,表达式,…	左到右	从左向右顺序运算

注：同一优先级的运算符，运算次序由结合方向所决定。

上表不容易记住。其实也用不着死记，用得多，看得多自然就记得了。也有人说不用记这些东西，只要记住乘除法的优先级比加减法高就行了，别的地方一律加上括号。这在你自己写代码的时候，确实可以，但如果是你去阅读和理解别人的代码呢？别人不一定都加上括号了吧？所以，记住这个表，我个人认为还是很有必要的。

2.9.2，一些容易出错的优先级问题

上表中，优先级同为 1 的几种运算符如果同时出现，那怎么确定表达式的优先级呢？这是很多初学者迷糊的地方。下表就整理了这些容易出错的情况：

优先级问题	表达式	经常误认为的结果	实际结果
.的优先级高于* ->操作符用于消除这个问题	*p.f	p 所指对象的字段 f (*p).f	对 p 取 f 偏移，作为指针，然后进行解除引用操作。*(p.f)
[]高于*	int *ap[]	ap 是个指向 int 数组的指针 int (*ap)[]	ap 是个元素为 int 指针的数组 int *(ap[])
函数()高于*	int *fp()	fp 是个函数指针，所指函数返回 int。 int (*fp)()	fp 是个函数，返回 int * int *(fp())
== 和!=高于位操作	(val & mask != 0)	(val & mask)!= 0	val & (mask != 0)
== 和!=高于赋值符	c = getchar() != EOF	(c = getchar()) != EOF	c = (getchar() != EOF)
算术运算符高于位移运算符	msb << 4 + lsb	(msb << 4) + lsb	msb << (4 + lsb)
逗号运算符在所有运算符中优先级最低	i = 1,2	i = (1,2)	(i = 1),2

这些容易出错的情况，希望读者好好在编译器上调试调试，这样印象会深一些。一定要多调试，光靠看代码，水平是很难提上来的。调试代码才是最长水平的。