

Datentypen		
Datentyp	Mindestwertebereich	typischer Wertebereich
char	-127 - 127	-128 - 127
short	-32.767 - 32.767	-32.767 - 32.767
int	-32.767 - 32.767	-2.147.483.648 - 2.147.483.647
long	-2.147.483.647 - 2.147.483.647	-2.147.483.647 - 2.147.483.647
long long	-9.223.372.036.8-54.775.807 - 9.223.372.036.8-54.775.807	-9.223.372.036.854.775.807 - 9.223.372.036.854.775.807
unsigned char	0-255	0-255
unsigned short	0 - 65.535	0 - 65.535
unsigned int	0 - 65.535	0 - 4.294.967.295
unsigned long	0 - 4.294.967.295	0 - 4.294.967.295
unsigned long long	0 - 18.446.744.0-73.709.551.615	0 - 18.446.744.073.709.551.615
float	$10^{-37} - 10^{37}$	$1.175494351 \cdot 10^{38} - 3.402823466 \cdot 10^{38}$
double	$10^{-37} - 10^{37}$	$2.2250738585072014 \cdot 10^{-308} - 1.7976931348623158 \cdot 10^{308}$
long double	$10^{-37} - 10^{37}$	$3.362103143112093506262677817321752602598 \cdot 10^{-4932} - 1.189731495357231765021263853030970205169 \cdot 10^{4932}$

Priorität von Operatoren	
Operator	Reihenfolge
[]	links nach rechts
()	
.	
->	
++ -- (postfix)	
{ }	rechts nach links
++ -- (prefix)	
sizeof	
~	
!	
- + (als Vorzeichen)	
& (Adressoperator)	
* (Variablenoperator)	
(type name) (Typkonvertierung)	
* / %	
+ -	links nach rechts
<< >>	links nach rechts
< > <= >=	links nach rechts
== !=	links nach rechts
& (bitweises UND)	links nach rechts
^ (bitweises XOR)	links nach rechts
(bitweises ODER)	links nach rechts
&& (logisches UND)	links nach rechts

Priorität von Operatoren (cont)

(logisches ODER)	links nach rechts
?:	rechts nach links
alle Zuweisungen wie =, +=, -=, ...	rechts nach links
,	links nach rechts
Oben höchste Priorität, unten niedrigste.	

Zusammengesetzte Zuweisungsoperatoren

Bezeichnung	Funktion
+=	Addition
-=	Subtraktion
*=	Multiplikation
/=	Division
%=	Modulo
<<=	Shift
Beispiel: x += 5; //Addiert 5 zur Variable x.	

andere zusammengesetzte Operatoren

Bezeichnung	Funktion
->	Elementzugriff
++	Inkrement (+1)
--	Dekrement (-1)
<<	Links-Schieben (Shift)
>>	Rechts-Schieben (Shift)
<=	kleiner gleich
>=	größer gleich
==	Geichheit
!=	Ungleichheit
&&	logisches UND
	logisches ODER

Konvertierung zwischen Datentypen

Umwandlung	Beispiel
explizite Typumwandlung	char c = 'A'; int i; i = (int) c;
Ziffernzeichen in Ziffer konvertieren	int i; i = (int) '5' - (int) '0';
Groß- in Kleinbuchstaben implizite Typumwandlung	char c = 'A'; c += 32;

Einzelne Operatoren

Bezeichnung	Funktion
!	logisches Nicht
~	bitweises NICHT
&	bitweises UND
	bitweises ODER
^	XOR
%	Modulo (Rest)
*	Multiplikation
-	Subtraktion
+	Addition
/	Division
<	kleiner
>	größer
=	Zuweisung
.	Elementzugriff
?	Bedingung

Wahrheitswerte (bool'sche Werte)

Werden für gewöhnlich als int dargestellt.

0 = falsch

alles ungleich null = wahr

