Equality operators:

```
==      equal to
!=      not equal to
```

They fall into a separate precedence group, beneath the relational operators and have the left-to-wright associativity.

Example:

```
i=1   j=2   k=3

Expression              Interpretation    Value
-------------------------------------------------
i < j                       true            1
(i + j) >= k                true            1
(j + k) > (i + 5)           false           0
k != 3                      false           0
j == 2                      true            1
```

Logical operators:

```
Operator    Meaning
-------------------
   &&         and
   ||         or
```

Example:

```
  i=7    f=5.5   c='w'

  Expression                  Interpretation     Value
  ------------------------------------------------
  ( i >= 6) && (c == 'w')        true            1
  ( i >= 6) || (c == 119)        true            1
  ( f < 11) && (i > 100)         false           0
  ( c == 'p') || ((i +f ) <= 10) true            1
```

Logical negation operator (logical not): `!`

Attention:

Complex logical expressions joined by the operators `&&` and `||` are evaluated lef-to-right, but only until the overall the true/false value has been established.

Assignment operators

The most commonly used assignment operator is `=` .

```
identifier = expression
```

If the two operands are of different types, then the right-hand-side expression wil automatically be converted to the type of the identifier on the left.

Multiple assignements are permissible:

```
identifier1 = identifier2 = ... = expression
```

The assignements are carried out from right to left.

Five additional assignement operators: `+=` `-=` `*=` `/=` `%=`.

The conditional operator

```
expression1 ? expression2 : expression3
```

Operator precedence groups:

Precedences and associativity of operators

Operators	Associativity
`() [] -> .`	L→R
`! - ++ -- + - *` & (type) sizeof	R→L
`* / %`	L→R
`+ -`	L→R
`<< >>`	L→R
`< <= > >=`	L→R
`== !=`	L→R
&	L→R
^	L→R
|	L→R
`&&`	L→R
`||`	L→R
`?:`	L→R
`= += -= *= /= %= ^= |= <<= >>=`	R→L
,	L→R

Library functions

Some commonly used library functions

Function	Type	Purpose
abs(i)	int	return the absolute value of i
ceil(d)	double	round up to the next integer value
cos(d)	double	return the cosine of d
cosh(d)	double	return the hyperbolic cosine of d
exp(d)	double	raise e to the power of d
fabs(d)	double	return the absolute value of d
floor(d)	double	round down to the next integer value
fmod(d1,d2)	double	return the remainder of $d1/d2$ (the same sign as $d1$)
getchar()	int	enter a character from the standard input
log(d)	double	return the natural logarithm of d
pow(d1,d2)	double	return $d1$ raised to the $d2$ power
printf(...)	int	send data items to the standard output
putchar(c)	int	send a character to the standard output
rand()	int	return a random positive integer
sin(d)	double	return the sign of d
sqrt(d)	double	return the square root of d
srand(u)	void	initialize the random number generator
scanf(...)	int	enter data items from the standard input
tan(d)	double	return the tangent of d
toascii(c)	int	convert value of argument to ASCII
tolower(c)	int	convert letter to lower case
toupper(c)	int	convert letter to uppercase