

### instruction sequences

\a - alert (bell) character

\b - backspace

\c - newcharacter

\f - formfeed

\n - newline

\r

=

carriage return

\t - horizontal tab

\v - vertical tab

\\ - backslash

\? - question mark

\' - single quote

\" - double quote

\ddd - octal bit pattern

\ooo - octal number

\xhh

=

hexadecimal number

\0 - nul

### relational operators

!= -

not equal to

==

=

is equal to

++

=

increment by one

||

=

OR

&&

—  
—

AND

basic data types (objects)

int

—  
—

integers

float

—  
—

floating point (with fractional part)

char

—  
—

character (a single byte)

short

—  
—

short integer

long

—  
—

long integer

double

—  
—

double precision floating point

printf conversion characters

%

—  
—

(each argument)

[%c](#) - single character

char

[%d](#) - integer decimal

-

int  
%ld - long integer decimal

-

int  
%e

=

float or double w/ exponent, e.g., [-]m.nnnnnnE[±]xx

float or double  
%f - floating

float or double  
%g

-  
-

e or f format, whichever is shorter

float or double  
%o - unsigned octal

int

%s

-

string

char[] or char \*  
%u - unsigned integer decimal

unsigned int  
%x - unsigned hexadecimal



int  
%% - % itself

%3d - print in a 3 digit field, right justified  
%3.0f - print no decimal point and no fraction  
%3.1f - print 1 digit after the decimal point  
%.1f - print 1 digit after the decimal point, any width

Between the % and the conversion character there may be a minus sign, to specify left adjustment of the field, and two digit strings separated by a period. The first string specifies minimum field width, and the second string specifies the maximum number of characters to be printed from the string.

```
:%10s:      :hello, world:
:%-10s:     :hello, world:
:%20s:      :      hello, world:
:%-20s:     :hello, world      :
:%20.10s:   :      hello, wor:
:%-20.10s:  :hello, wor      :
:%.10s:     :hello, wor:
```

### scanf conversion characters

%d

decimal integer

int \*  
%o

octal integer

int \*  
%x

hexadecimal integer

int \*  
%h

short decimal integer

short int \*  
%c

character

char \*  
[%s](#)

string

char[] or char \*  
[%f](#)

floating-point number

float \*  
[%e](#)

same as %f

float \*

C Hierarchy (K & R, p. 49)

() [] -> .	L to R
! ~ ++ -- - (type) * & sizeof	R to L
* / %	L to R
+ -	L to R
<< >>	L to R
< <= > >=	L to R
== !=	L to R
&	L to R
^	L to R
	L to R
&&	L to R
	L to R
?:	R to L
= += -= etc.	R to L
,	L to R